



**Calhoun: The NPS Institutional Archive**  
**DSpace Repository**

---

Theses and Dissertations

Thesis and Dissertation Collection

---

1976-06

# Computer aided control system design using frequency domain specifications.

Mancini, Anthony J.

Monterey, California. Naval Postgraduate School

---

<http://hdl.handle.net/10945/17699>

*Downloaded from NPS Archive: Calhoun*



Calhoun is a project of the Dudley Knox Library at NPS, furthering the precepts and goals of open government and government transparency. All information contained herein has been approved for release by the NPS Public Affairs Officer.

**Dudley Knox Library / Naval Postgraduate School**  
**411 Dyer Road / 1 University Circle**  
**Monterey, California USA 93943**

<http://www.nps.edu/library>

COMPUTER AIDED CONTROL SYSTEM DESIGN USING  
FREQUENCY DOMAIN SPECIFICATIONS

Anthony J. Mancini

# NAVAL POSTGRADUATE SCHOOL

Monterey, California



## THESIS

COMPUTER AIDED CONTROL SYSTEM DESIGN USING  
FREQUENCY DOMAIN SPECIFICATIONS

by

Anthony J. Mancini

June 1976

Thesis Advisor:

Dr. G. J. Thaler

Approved for public release; distribution unlimited.

T174159

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Computer Aided Control System Design Using Frequency Domain Specifications		5. TYPE OF REPORT & PERIOD COVERED Engineer's Thesis (June 1976)
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s)  Anthony J. Mancini		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS  Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS  Naval Postgraduate School Monterey, California 93940		12. REPORT DATE June 1976
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)  Naval Postgraduate School Monterey, California 93940		15. SECURITY CLASS. (of this report)  UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report)  Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Control System Design                      Transfer Function Synthesis Computer Aided Control System Design Series Compensation                      Frequency Response Techniques Linear Control Systems Computer Aided Design		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The primary intent of this work is to investigate computer aided compensator design using classical frequency response techniques in conjunction with the techniques of modern mathematical programming. A computer program to perform the automated design task is presented. This particular algorithm is based on the constrained optimization technique introduced by M. J. Box. In particular, the desired open loop frequency response is		



20. specified for a number of discrete frequency points over the frequency range of interest. Then the minimization routine is used to vary the compensator parameters in such a manner as to minimize a cost functional based on the difference between the actual and desired open loop frequency response of the compensated system. To illustrate the algorithm several detailed examples using the program are presented.

COMPUTER AIDED CONTROL SYSTEM DESIGN USING FREQUENCY  
DOMAIN SPECIFICATIONS

by

Anthony Joseph Mancini  
Lieutenant United States Navy  
E.S.E.-E., University of Oklahoma, 1970  
M.S.E.E., Naval Postgraduate School, 1975

Submitted in partial fulfillment of the  
requirements for the degree of

ELECTRICAL ENGINEER

from the  
NAVAL POSTGRADUATE SCHOOL  
June 1976

## ABSTRACT

The primary intent of this work is to investigate computer aided compensator design using classical frequency response techniques in conjunction with the techniques of modern mathematical programming. A computer program to perform the automated design task is presented. This particular algorithm is based on the constrained optimization technique introduced by M. J. Ecx.

In particular, the desired open loop frequency response is specified for a number of discrete frequency points over the frequency range of interest. Then the minimization routine is used to vary the compensator parameters in such a manner as to minimize a cost functional based on the difference between the actual and desired open loop frequency response of the compensated system. To illustrate the algorithm several detailed examples using the program are presented.

## TABLE OF CONTENTS

LIST OF FIGURES.....	7
ACKNOWLEDGEMENTS.....	12
I.    INTRODUCTION .....	13
II.   THE AUTOMATED DESIGN PROBLEM .....	16
A.   GENERAL AUTOMATED DESIGN PHILOSOPHY .....	16
B.   SOME ASPECTS OF COMPUTERIZING THE CLASSICAL DESIGN OF CONTROL SYSTEMS .....	18
C.   A FORMULATION OF THE AUTOMATIC DESIGN PROBLEM IN THE FREQUENCY DOMAIN .....	20
III.  THE MINIMIZATION TECHNIQUE USED .....	26
A.   GENERAL .....	26
B.   DESCRIPTION OF THE ALGORITHM .....	28
IV.   PROGRAM DESCRIPTION AND UTILIZATION .....	32
A.   DESCRIPTION OF ALGORITHM IMPLEMENTATION ....	32
B.   DATA INPUT AND OUTPUT .....	38
C.   SERIES COMPENSATOR DESIGN EXAMPLE PROBLEMS .	49
1.   Compensator Design, Example 1 .....	49
2.   Compensator Design, Example 2 .....	63
3.   Compensator Design, Example 3 .....	75
4.   Compensator Design, Example 4 .....	94
5.   Compensator Design, Example 5 .....	113
V.    SYNTHESIS OF TRANSFER FUNCTIONS FROM FREQUENCY RESPONSE DATA .....	125
A.   GENERAL DISCUSSION .....	125
B.   TRANSFER FUNCTION SYNTHESIS: EXAMPLE PROBLEMS .....	129
1.   Synthesis, Example 1 .....	132
2.   Synthesis, Example 2 .....	151
3.   Synthesis, Example 3 .....	162
4.   Synthesis, Example 4 .....	172

VI. CONCLUSION AND SUGGESTED FUTURE STUDIES .....	195
Appendix A: DESCRIPTION OF SUBROUTINES.....	198
Appendix B: PROGRAM LISTING.....	214
Appendix C: DESCRIPTION OF FORTRAN VARIABLE NAMES.....	293
LIST OF REFERENCES.....	299
INITIAL DISTRIBUTION LIST.....	301

# LIST OF FIGURES

II-1	General System, Block Diagram.....	22
IV-1	Program Block Diagram.....	34
IV-2	Example Data Deck.....	48
IV-3	Design Example 1, Block Diagram.....	50
IV-3A	Unccompensated Open Loop System Bode Plot.....	53
IV-3B	Compensated System Magnitude and Phase Responses.....	54
IV-3C	Magnitude vs. Phase.....	55
IV-3D	Computer Numerical Output Example 1.....	56
IV-3E	Computer Numerical Output Example 1.....	57
IV-3F	Computer Numerical Output Example 1.....	58
IV-3G	Magnitude and Phase Difference Plots.....	59
IV-3H	Example 1, Run #2 Magnitude and Phase Plots.....	60
IV-3I	Example 1, Run #2 Magnitude vs. Phase.....	61
IV-3J	Example 1, Run #2 Magnitude and Phase Difference Plots.....	62
IV-4	Design Example 2, Block Diagram.....	64
IV-4A	Unccompensated Open Loop System Bode Plot.....	66
IV-4B	Magnitude vs. Frequency.....	67
IV-4C	Phase vs. Frequency.....	68
IV-4D	Magnitude vs. Phase.....	69
IV-4E	Computer Numerical Output Example 2.....	70
IV-4F	Computer Numerical Output Example 2.....	71
IV-4G	Computer Numerical Output Example 2.....	72
IV-4H	Magnitude Difference vs. Frequency.....	73
IV-4I	Phase Difference vs. Frequency.....	74
IV-5	Design Example 3, Block Diagram.....	75
IV-5A	Magnitude vs. Frequency.....	78
IV-5B	Phase vs. Frequency.....	79



IV-5C	Magnitude vs. Phase.....	80
IV-5D	Computer Numerical Output Example 3, Run #1.....	81
IV-5E	Computer Numerical Output Example 3, Run #1.....	82
IV-5F	Computer Numerical Output Example 3 Run #1.....	83
IV-5G	Magnitude Difference vs. Frequency.....	84
IV-5H	Phase Difference vs. Frequency.....	85
IV-5I	Magnitude vs. Frequency.....	86
IV-5J	Phase vs. Frequency.....	87
IV-5K	Magnitude vs. Phase .....	88
IV-5L	Computer Numerical Output Example 3, Run #2.....	89
IV-5M	Computer Numerical Output Example 3, Run #2.....	90
IV-5N	Computer Numerical Output Example 3, Run #2.....	91
IV-5O	Magnitude Difference vs. Frequency.....	92
IV-5P	Phase Difference vs. Frequency.....	93
IV-6	Design Example 4, Block Diagram.....	96
IV-6A	Uncompensated System Bode Plot.....	97
IV-6B	Magnitude vs. Frequency, Single Section Compensator.....	98
IV-6C	Phase vs. Frequency, Single Section Compensator.....	99
IV-6D	Computer Numerical Output.....	100
IV-6E	Computer Numerical Output.....	101
IV-6F	Computer Numerical Output.....	102
IV-6G	Magnitude Difference vs. Frequency, Single Section Compensator .....	103
IV-6H	Phase Difference vs. Frequency, Single Section Compensator.....	104
IV-6I	Magnitude vs. Frequency, Double Section Compensator.....	105

IV-6J	Phase vs. Frequency, Double Section Compensator.....	106
IV-6K	Magnitude vs. Phase, Double Section Compensator.....	107
IV-6L	Computer Numerical Output.....	108
IV-6M	Computer Numerical Output.....	109
IV-6N	Computer Numerical Output.....	110
IV-6O	Magnitude Difference vs. Frequency, Double Section Compensator.....	111
IV-6P	Phase Difference vs. Frequency, Double Section Compensator.....	112
IV-7	Unscaled System Block Diagram.....	115
IV-7A	Scaled System Block Diagram.....	116
IV-7B	Magnitude vs. Frequency.....	117
IV-7C	Phase vs. Frequency.....	118
IV-7D	Magnitude vs. Phase.....	119
IV-7E	Computer Numerical Output.....	120
IV-7F	Computer Numerical Output.....	121
IV-7G	Computer Numerical Output.....	122
IV-7H	Magnitude Difference vs. Frequency.....	123
IV-7I	Phase Difference vs. Frequency.....	124
V-1	Transfer Function Synthesis Data Deck.....	131
V-2	Input Frequency response Data.....	134
V-2A	Magnitude vs. Frequency, Run #1.....	135
V-2B	Phase vs. Frequency, Run #1.....	136
V-2C	Magnitude vs. Phase, Run #1.....	137
V-2D	Magnitude Difference vs. Frequency, Run #1...	138
V-2E	Phase Difference vs. Frequency, Run #1.....	139
V-2F	Computer Numerical Output, Run #1.....	140
V-2G	Computer Numerical Output, Run #1.....	141
V-2H	Computer Numerical Output, Run #1.....	142
V-2I	Magnitude vs. Frequency, Run #2.....	143
V-2J	Phase vs. Frequency, Run #2.....	144
V-2K	Magnitude vs. Phase, Run #2.....	145
V-2L	Magnitude Difference vs. Frequency, Run #2...	146
V-2M	Phase Difference vs. Frequency, Run #2.....	147

V-2N	Ccputer Numerical Output, Run #2.....	148
V-2O	Ccputer Numerical Output, Run #2.....	149
V-2P	Ccputer Numerical Output, Run #2.....	150
V-3	Input Frequency Response Data.....	153
V-3A	Magnitude vs. Frequency.....	154
V-3B	Phase vs. Frequency.....	155
V-3C	Magnitude vs. Phase.....	156
V-3D	Magnitude Difference vs. Frequency.....	157
V-3E	Phase Difference vs. Frequency.....	158
V-3F	Ccputer Numerical Output.....	159
V-3G	Ccputer Numerical Output.....	160
V-3H	Ccputer Numerical Output.....	161
V-4	Input Frequency Respcnse Data.....	163
V-4A	Magnitude vs. Frequency.....	164
V-4B	Phase vs. Frequency.....	165
V-4C	Magnitude vs. Phase.....	166
V-4D	Magnitude Difference vs. Frequency.....	167
V-4E	Phase Difference vs. Frequency.....	168
V-4F	Ccputer Numerical Output.....	169
V-4G	Ccputer Numerical Output.....	170
V-4H	Ccputer Numerical Output.....	171
V-5A	Magnitude vs. Frequency, Run #1.....	174
V-5B	Phase vs. Frequency, Run #1.....	175
V-5C	Magnitude vs. Phase, Run #1.....	176
V-5D	Magnitude Difference vs. Frequency, Run #1...	177
V-5E	Phase Difference vs. Frequency, Run#1.....	178
V-5F	Ccputer Numerical Output, Run #1.....	179
V-5G	Ccputer Numerical Output, Run #1.....	180
V-5H	Computer Numerical Output, Run #1.....	181
V-5I	Magnitude vs. Frequency, Analytical Transfer Function.....	182
V-5J	Phase vs. Frequency, Analytical Transfer Function.....	183
V-5K	Magnitude vs. Phase, Analytical Transfer Function.....	184

V-5L	Magnitude Difference vs. Frequency, Analytical Transfer Function.....	185
V-5M	Phase Difference vs. Frequency, Analytical Transfer Function.....	186
V-5N	Magnitude vs. Frequency, Run #2.....	187
V-5O	Phase vs. Frequency, Run #2.....	188
V-5P	Magnitude vs. Phase, Run #2.....	189
V-5Q	Magnitude Difference vs. Frequency, Run #2...	190
V-5R	Phase Difference vs. Frequency, Run #2.....	191
V-5S	Computer Numerical Output, Run #2.....	192
V-5T	Computer Numerical Output, Run #2.....	193
V-5U	Computer Numerical Output, Run #2.....	194

## ACKNOWLEDGEMENTS

The author would like to express his sincere thanks and appreciation to Dr. G. J. Thaler for his continued encouragement, advice, and unselfish sharing of professional expertise during the research and development of this thesis.

A word of thanks is also owed to three members of the Naval Postgraduate School Computer Center Operations Staff, Mr. Ed Dornellan, Mrs. Kris Butler and Mr. M. Anderson, whose professional services contributed extensively to the efficient use of the time necessary in performing the large amount of computer work associated with the development, debugging, and testing of the program presented.

Finally, to my wife and children goes a special debt of gratitude for their patience, understanding, and encouragement during the months of development and research of this thesis. It would not have been possible to accomplish this undertaking without their continued help throughout this period.

## I. INTRODUCTION

Concurrent with the proliferation of the digital computer as a viable, indeed often indispensable, engineering tool, the development of state-space methods resulted in the application of the digital computer in the area of modern control theory over the past decade and a half. Initially this application was primarily of an analysis nature, but as the theory of optimal control came into wider publication and acceptance, the role of the digital computer began to take on aspects of the design function. This situation is graphically illustrated by the myriad of industrial proprietary and university developed computer programs for computer aided control system design based upon state variable and optimal control theories. However, as pointed out by Mitchell [11] comparatively little work has been done to extend and apply the computational power, speed, and accuracy of modern digital computers to the proven classical theory and methods of control system design. After this essential abandonment of the classical methods (particularly by the academic and theoretical factions of the control engineering community) in favor of the modern theory and techniques, recent publications indicate a realization of the need that both approaches to system design be used in a complimentary rather than mutually exclusive manner. In particular the works of Coffey, MacFarlane, Mitchell, Page, Rosenbrock, Stear, [5,8,11,14] and others show a renewed interest in the tried and proven classical theory of design coupled with the new techniques and approaches developed by proponents of modern control theory, as applicable to the development of usable computer aided control system design algorithms.



Control system design based on frequency domain specifications, for linear time invariant parameter systems using classical methods, has traditionally been done essentially using trial and error techniques. That is, once the form and location of the compensator has been decided upon, the design process reduces to one of iterative analysis until the specifications are met or it becomes obvious to the designer that the compensator form chosen will not satisfy the specifications. When systems of any complexity are involved this method quite often transforms into an extremely tedious and time consuming process.

The availability of a good analysis program coupled with a digital computer operating in an interactive mode, to provide minimal turn around time, can be used to somewhat minimize the tedium of this type of design procedure. By varying the compensator parameters the designer may then observe the effects upon system performance without having to expend considerable effort in repeated calculations. This design by iterative analysis may be continued allowing the computer to perform the calculations until the designer is satisfied with the response obtained. With the recent refinements in mathematical programming techniques, it is possible to relegate some of the design decisions directly to the computer program, thus even further minimizing the time and efforts required of the design engineer. The ultimate goal, which is of course at present a long way down the road, would be to input only the plant dynamics and the desired specifications and have the computer perform the design process completely, without any interaction with the design engineer.

In this thesis an investigation of the automation of the design procedure is undertaken. In particular, given the desired open loop response in the frequency domain a minimization routine is used in conjunction with an analysis

algorithm to systematically vary the compensator parameters to approximate the desired response in accordance with a performance measure which indicates the deviation of the actual open loop frequency response from the desired open loop frequency response.

## II. THE AUTOMATED DESIGN PROBLEM

### A. GENERAL AUTOMATED DESIGN PHILOSOPHY

The word design, as used in many engineering texts of various disciplines, runs the gamut of describing a process dealing essentially with scaling already existing configurations and components to one dealing with a highly creative activity allowing a maximum amount of freedom in satisfying a set of requirements or specifications. On a day to day basis the majority of the design work accomplished by engineers lies somewhere between these two extremes, combining elements of relatively routine, straight forward calculations with those of creativity and imagination.

The overall scope of the design process in general is indeed so formidable that it often defies complete description in other than general terms, much less quantification in precise mathematical language. This latter is unfortunately a necessity in achieving the automation of this process within the limitations of the presently available hardware and software accessible by the engineer. Largely for this reason, when considering the automatic design problem it must by necessity be viewed, for the near future at least, in terms of a limited scope. That is, such items as creativity, experience, and intuition, which are undoubtedly elements of the total design process as viewed in its entirety, cannot be achieved by a computer as the device exists at the present time.

One of the key elements of any design process is decision making. In the overall design process the engineer makes decisions based upon the information available to him at the time of the design undertaking. Quite often these decisions are based on a comparison of numerical values of various elements of the design configuration. It is in this specific area of the overall design process that the modern day computer can serve as a significant tool for the engineer involved in design, particularly when a large number of comparisons (expressable in numerical terms) are involved. In other words, the computer can be used in a combination analysis and elementary decision making role to aid the engineer in the overall design procedure. By means of analysis numerical values may be obtained, forming a basis from which comparisons can be drawn. The analysis portion is generally essential for exact design and forms the basis for any optimization. The speed available in the digital computer allows many calculations and comparisons, over a wide range of possible solutions, to be performed which would be prohibitive if performed by hand.

The employment of the digital computer as an aid in the aggregate design process is perhaps as much an art as the design process itself. While the use of computer aided design programs can free the engineer from many burdensome and time consuming calculations, the individual engineer must still provide the framework in which to interpret the results in the hard light of physical significance. The engineer must also be capable and willing to define the problem in a form with which the computer can work. More often than not, this latter requires a much sharper, clearer definition (and subsequently understanding) of the problem than if the same problem were being designed long hand. Unlike the analysis problem, which generally possesses a specific answer, the design problem may not have a unique solution which further complicates any computer aided design

algorithm. Another aspect that should not be overlooked is the fact that when using a computer aided design program the engineer often receives, if nothing else, an indication of the solutions which are not feasible. This in itself is often not a trivial result especially in dealing with systems of any complexity.

Another important factor which should not be overlooked is that once an answer is obtained using a computer aided design algorithm some type of sensitivity study will more than likely be necessary. This is largely due to the fact that component values in physical hardware cannot generally be controlled to the precision with which a computer does its calculations. Therefore, while a particular algorithm for use in automated design may generate what is indeed a theoretically plausible solution the implementation of such a solution may not be possible if it is extremely sensitive to the parameter values involved.

## B. SOME ASPECTS OF COMPUTERIZING THE CLASSICAL DESIGN OF CONTROL SYSTEMS

Classically the purpose of the design problem in feedback control systems has been to make a given plant satisfy a set of performance specifications imposed upon the system. Design of linear, time invariant compensators for the control of linear systems has formed a major portion of the design efforts of conventional servomechanisms. In the process of investigating the automation of this design procedure, certain key elements were found to be generally necessary in implementing a basic practical computer algorithm for automated design, regardless of whether it is based on time or frequency domain approaches. These elements consisted of:

1. A complete description of the plant or process to be controlled.
2. Specifications describing the response desired from the system.
3. Some type of criterion function or performance measure relating the actual to the desired response.
4. The form or structure of the compensator or control to be implemented.

These items appear in varying degrees as a common denominator among present day automated design algorithms [1,5,11,13] regardless of the specific methods employed or systems being considered. That is to say, these elements appear in algorithms designed for time domain, transform domain, stochastic processes, and nonlinear systems. Generally items 1. and 4. are employed in some form of analysis algorithm while items 2. and 3. form the basis for a minimization (or maximization) routine that provides a foundation for some type of decision making within the algorithm. This idea of using a performance measure upon which to base some type of decision making routine is a direct carryover from developments in the computerized design of optimal control systems which relies heavily upon the idea of minimizing some criterion function. Unlike the techniques of optimal control however, the cost functional as used in this context is not necessarily a function of the states and the applied control, but rather a measure of the deviation of the actual response from some desired response which has been specified by the designer. This is perhaps a fine distinction but one that must be recognized and kept in mind. One of the many consequences of this is that there is no longer a strong motivation to restrict the performance measure to a quadratic form in order that the second method of Liapunov may be used to prove that a stable system will result. Rather, the form of the cost function in computerized design based upon classical control theory is chosen to have properties which aid in the convergence of



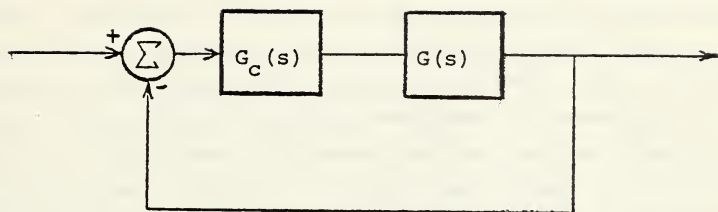
the numerical minimization routine used in the algorithm or which has some inherent physical significance in the design process.

### C. A FORMULATION OF THE AUTOMATIC DESIGN PROBLEM IN THE FREQUENCY DOMAIN

In this thesis the automatic design problem based on theory developed for classical design methods using frequency domain procedures is investigated. Several approaches based on frequency domain techniques have been presented by various people to automate the control system design process [5,11,13,14,20]. These algorithms range from methods designed primarily for use with multivariable systems to specialized interpretations of frequency domain data such as vector frequency domain techniques and inverse Nyquist methods. The suitability of these various methods have been demonstrated by their respective authors and others working in the area of computer aided control system design. While each represents a step forward in the line of progress toward automating the design process using frequency domain techniques the suitability of each algorithm is dictated to some extent by the particular type of design problem being solved. Many of these methods require gradient calculations, and those that do not rely primarily on simplex nonlinear programming algorithms. The former methods restrict the form of the cost function to be minimized in the sense that it must be differentiable and problems inherent with numerical differentiating algorithms are obviously involved. The latter methods involving simplex algorithms have difficulty in handling constrained minimization problems, particularly when solutions exist near or on the constraint boundaries.

One of the advantages of the frequency response approach lies in the fact that it lends itself easily to the use of open-loop frequency response data in designing closed loop systems. The use of frequency response methods provides a graphical portrayal of the system response that lends itself well to quick interpretation by the designer. Also since approaches such as those developed by Bode were intended to give simple approximate methods for use in hand calculations, adaptation of these methods in a computer algorithm allows the engineer to plan his design using such techniques and then employ the aid of the computer without the requirement of significant problem restructuring in order to cast the problem in some particular format not generally familiar to the designer, but necessary for solution by a computer algorithm. With systems involving complex transfer functions such restructuring can itself become quite burdensome thus defeating one of the primary reasons for turning to computer aided design. Perhaps it should be pointed out at this point that even using a computer algorithm to aid in the design process does not totally relieve the engineer of all hand calculations. Time must be taken to carefully map out a plan of attack on a particular problem. In essence the designer must thoroughly understand the problem and this is generally achieved by some preliminary rough sketches and calculations before the computer ever makes its appearance in the design process.

Throughout this thesis, the vehicle used for presenting the automated design algorithm is the single-input single-output unity feedback system of the form shown in figure II-1.



General System, Block Diagram

figure II-1

The plant is considered to be linear, time invariant, and completely known. Thus the plant may be described by a rational transfer function in the Laplace variable  $s$  of the form

$$G(s) = \frac{\prod_{i=0}^n (s + z_i)}{\prod_{i=0}^m (s + p_i)}$$

where  $z_i$  and  $p_i$  may be complex constants. The compensator  $G_c(s)$  is also considered to be a rational transfer function in  $s$ .

The specifications are given in terms of the system open loop frequency response, and the desired response is achieved by reshaping the system open loop frequency response through the use of the compensator in the feed forward path. This basic procedure is well known to most control engineers and is by no means a novel approach to the design problem [3,10,17,18], but through the years it has found extensive use in the practical design of control systems.

The method proposed and under investigation here is a relatively straight forward adaptation of this technique with a computer algorithm harnessed to do the compensator parameter variation in order to systematically reshape the system open loop frequency response with a minimum amount of calculation and iteration required on the part of the designer. The desired open loop response is specified by means of a magnitude and phase profile over the frequency range of interest. This is nothing more than a set of desired magnitude and corresponding phase values at discrete frequency points selected by the designer. In addition a specific form or structure is postulated for the compensator transfer function  $G_c(s)$ . At this point of the automated design process the algorithm relies heavily on the experience and intuition of the individual designer in choosing a structure or form of compensator that will have some reasonable probability of success in reshaping the frequency response curves in order to satisfy the design specification. Since no firm analytical basis exists for the determination of a specific structure for the compensator, the computer algorithm must be given this information or an iterative method based on some type of criterion must be used in order to establish the form. This latter method was felt to be much too time consuming and restrictive, because of the wide range of variations in possible structures that might have to be considered, and essentially beyond the scope of this investigation. Thus the choice of the compensator form is relegated to the program user as a design decision that must be undertaken prior to using the algorithm to determine specific values for the coefficients of the compensator transfer function that will yield the desired response. This procedure should not present a major stumbling block for the designer provided that he possesses a thorough understanding of the problem to be solved. In fact, by inputting the specific form

of the compensator the designer may investigate compensator configurations which might otherwise be considered unfeasible for the solution of the problem if some type of iterative algorithm were used in determining the structure. With these items forming the basic inputs to the algorithm a nonlinear performance measure coupled with a minimization routine is then used to minimize the difference between the desired and actual frequency response of the open loop transfer function given by,

$$T(s) = G_c(s) \cdot G(s)$$

The performance measure (or cost function) selected is of a normalized form primarily as a matter of convenience in interpreting the algorithm's results regardless of the particular problem being considered. That is, the cost function has a general form given by

$$J(j\omega) = f(1.0 - \frac{A(j\omega)}{D(j\omega)})$$

where  $D(j\omega)$  represents the desired open loop frequency response of the entire system,  $A(j\omega)$  is the actual open loop response, and  $\omega$  represents the set of discrete frequency values at which the cost function is to be evaluated. As mentioned previously this general form of normalized cost function was selected primarily as a matter of convenience in that a particular cost function can be constructed in such a manner that the optimal or smallest value of  $J(j\omega)$  under ideal solution of the problem will be zero.

By inputting the desired response in the form of the magnitude and phase profile described above, normal frequency domain specifications such as gain margin, phase margin, and bandwidth (here defined as the open loop gain crossover frequency) may be supplied to the algorithm in one type of format. With these specifications forming part of the gain and phase profiles they are automatically

incorporated into the overall cost function each time the algorithm evaluates the frequency response of the open loop system over the range of frequencies specified by the designer, thus also eliminating the requirement of specialized computation within the algorithm to check for satisfaction of these particular specifications as the compensator parameters are varied.



### III. THE MINIMIZATION TECHNIQUE USED

#### A. GENERAL

A variety of functional minimization techniques have come into use in recent years are discussed in ref. [8]. The minimization algorithm used in this work was developed originally by M. J. Fox, [4], and is capable of finding the minimum of a general nonlinear cost function, composed of several variables, within a constrained region. The complex method of Fox has been implemented in program form as part of the standard subroutine library at the Naval Postgraduate School W. F. Church Computer Center. The subroutine (BCXPLX), was originally programmed by R. R. Hilleary of the staff of the Naval Postgraduate School Computer Center and has subsequently been modified slightly by the author in order to render it more suitable for use in the scheme of the compensator optimization program.

In addition to its availability as a standard subroutine, several other features of this particular minimization algorithm made it an attractive choice for use in minimizing a cost function based on frequency domain specifications. While admittedly more efficient and sophisticated algorithms for function minimization are available, the flexibility and ease of programming of the complex method of Fox are among considerations that should not be overlooked. This method does not require that the derivatives of the cost function or implicit constraints be calculated, thus avoiding the sundry problems typically

associated with numerical methods for the calculation of derivatives, and eliminating the restriction that the derivative of the cost function exist over the range of interest. In implementing the minimization method the cost function need not be expressed explicitly in terms of the variables to be manipulated by the program in the minimization process. Unlike many of the other minimization programs available, BOXPLX, also provides a restart capability. That is, once a possible minimum value of the cost functional is reached the program automatically restarts from randomly generated initial values of the variables. While this method by no means guarantees that a global minimum will be achieved for every conceivable problem the restart capability does at least somewhat attempt to avoid the problem of the minimization terminating on a local minima within the search area. This method is also to some extent scale independent in that the size of the initial geometric figure used in the search for the minimum value of the objective function is roughly scaled to the order of the problem variables, through the use of the difference between the lower and upper bounds on the variables.

As is the case with other minimization algorithms, this particular one is not void of inherent disadvantages and difficulties. Present in any minimization algorithm of course is the recurring problem that no guarantee can be made of achieving a global minimum for all classes of problems. The complex method cannot handle equality constraints without modification of the algorithm. Also as the number of variables increases the method rapidly becomes inefficient. Finally the unconstrained problem is apparently more efficiently handled via gradient techniques.

## B. DESCRIPTION OF THE ALGORITHM

The minimization method developed by Ecx has its basis in the sequential simplex techniques of linear programming, but is specifically designed to obtain solutions to problems of constrained optimization and to avoid such problems as the inability of constructing the first simplex. Thus the resulting solution of the minimization problem is a set of variables which may lie at the extreme edges or boundaries of their permissible ranges. The problems generally associated with constraints on the variables is essentially attacked by using a flexible geometric figure having  $n+1$ , or more, vertices and capable of expanding or contracting, in any or all directions. Therefore, unlike many of the simplex algorithms no provision is made to maintain a regular geometry between the various points in an  $n$ -dimensional space, but rather a changing structure (dubbed a complex), capable of flattening, rounding corners, and eventually collapsing on the point representing the minimum of the objective function, is employed rendering the method more flexible in handling constrained minimization problems. The number of independent variables in the objective function and constraint equations dictate either directly or implicitly the dimensionality of the space to be searched. The vertices making up the complex are continually rejected and generated as a search for the minimum of the objective function is carried out with each new vertex generated being required to satisfy all the imposed constraints.

The complex method searches for the minimum of an objective function  $J(\underline{x})$ , where the vector  $\underline{x}$  represents a set of  $n$  variables, within a region of space bounded by upper and lower limits on the variables (explicit constraints) given by,

$$L_i \leq x_i \leq U_i, \quad i=1,2,3,\dots,n$$

and in addition subject to the restrictions imposed by the constraint functions (implicit constraints) of the form,

$$h_j(\underline{x}) \geq 0 \quad j=1,2,3,\dots,m.$$

As with the simplex methods this is an iterative algorithm. It uses a set of  $k \geq n+1$  points which form the vertices of the complex and simultaneously satisfy all the imposed constraints. Initially only one point or vertex must satisfy all the constraints and serves as a means of generating the other  $k-1$  vertices through the use of pseudorandom numbers  $r_i$ , uniformly distributed over the closed interval from zero to one. The additional  $k-1$  vertices of the first complex are established by the following formula:

$$x_i = L_i + r_i \cdot (U_i - L_i)$$

As can be seen from the formula, the vertices generated in this fashion will always satisfy the explicit constraints, however, each one must be checked to insure compliance with the implicit constraints. If an implicit constraint violation is indeed found to occur at this stage then the trial vertex is simply moved, repeatedly if necessary, halfway in toward the centroid of the other already accepted vertices, until ultimately a feasible point is found. Continued application of this procedure will result in the generation of the  $k-1$  additional points required for the initial complex within the feasible region defined by the constraints.

Once the initial complex has been determined the iteration proceeds by evaluating the objective function  $J(\underline{x})$  at each vertex. The vertex at which the objective function has the largest value is then designated as the current worst vertex and this point is reflected through the centroid of the remaining vertices establishing a new

complex. If this worst vertex is designated as  $\underline{x}^W$  then the overreflection is accomplished according to the formula

$$\underline{x}^N = (1.0 + \alpha) \cdot \underline{x}^{CR} - \alpha \cdot \underline{x}^W$$

where  $\alpha \geq 1.0$  is the overreflection coefficient,  $\underline{x}^{CR}$  represents the centroid of the remaining vertices, and  $\underline{x}^N$  is the new vertex. If the overreflection of the current worst vertex results in a violation of the constraints or if the new point still has the worst value in the set of vertices then the point is moved halfway toward the centroid used in the overreflection process. This retraction is repeated (if necessary) until the overreflection coefficient is reduced to some small value, delta, or a suitable new vertex is established. If the overreflection coefficient is reduced to the value delta and the objective function value at this new point is still the worst in the set of vertices then the projected vertex is replaced by its original value and the second worst vertex will be overreflected instead. This process keeps the complex moving toward the minimum of the objective function provided the figure has not collapsed into its centroid. Once assured that the new complex satisfies all the constraints and results in an improvement of the objective function, vertices of the complex are rejected and generated in a systematic manner aimed at minimization of the objective function. Thus essentially the complex moves over the feasible region eventually straddling and collapsing upon the point that renders a minimum value for the objective function.

This process continues and is eventually terminated when the complex shrinks to a predetermined acceptable small value epsilon given by:

$$\left\{ \frac{1}{k} \sum_{l=1}^k [ J(\underline{x}^C) - J(\underline{x}^V) ]^2 \right\}^{0.5} < \text{epsilon}$$

Where  $J(\underline{x}^C)$  represents the value of the objective function at the centroid and  $J(\underline{x}^V)$  is the value at the various vertices.

As pointed out by Box [4], the exact values of the overreflection coefficient, alpha, and the exact number of vertices, k, do not appear to be critical to the functioning of the algorithm provided that they are greater than unity and n+1 respectively. The main concern here is that a value of alpha larger than unity keeps the complex from shrinking prematurely as it traverses the feasible region in search of the minimum objective function value. Values of alpha = 1.3 and k = 2n as given in ref. 4 are used here since the exact values chosen are apparently not critical and good results have been reported, by the originator of the algorithm, using these particular values. For a detailed example of the iteration process and a more sophisticated explanation of the algorithm the interested reader is directed to refs. 2, 4, and 6.



#### IV. PROGRAM DESCRIPTION AND UTILIZATION

##### A. DESCRIPTION OF ALGORITHM IMPLEMENTATION

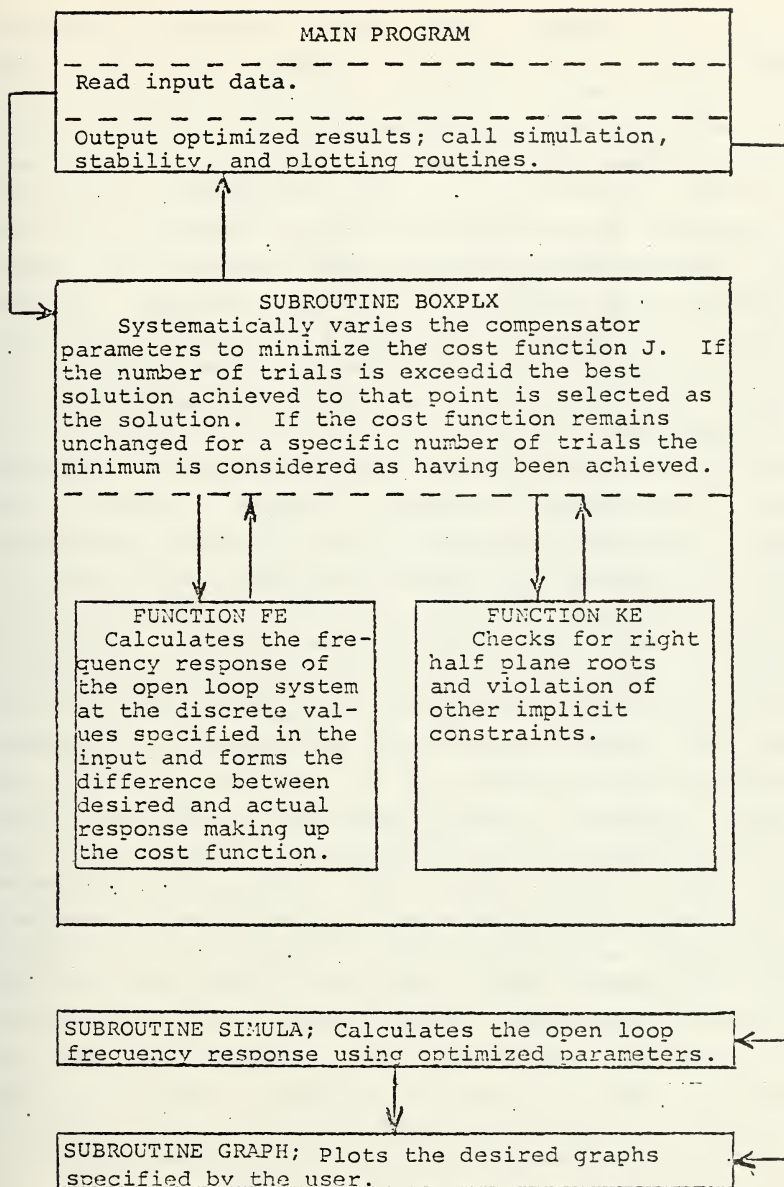
The frequency domain algorithm described in the latter part of Chapter II has been implemented in program form using FORTRAN IV coding and configured for use on the IBM 360-67 computer system available at the Naval Postgraduate School Computer Center. With the exception of the graphing subroutine the remainder of the program is self contained and may be used on any computer system having a standard FORTRAN IV compiler. Without the overhead routines required of the graphing subroutine, the basic program requires approximately 180k bytes of core memory, and including the core requirements of the plotting routines roughly 210k bytes of memory are required by the program. While admittedly less core would be necessary if such techniques as array overlaying were employed when programming the algorithm, it was felt that for simplicity in following the flow of the program by future users these techniques were best left unused.

The computer aided linear time invariant compensator optimization program (CALICO) is essentially composed of four main parts with numerous additional small subroutines functioning in supportive roles. Since often a program user may wish to perform certain modifications to programs of this nature in order that they may better serve his particular purpose, Appendix A provides a description of the supportive subroutines used within the program to facilitate



accomplishment of any program modifications that may be desired. A block diagram of the major components of the overall program are shown in figure IV-1. The main program serves primarily to input data describing the system under consideration and to output the values of the roots and the gain of the compensator that will best achieve the desired response in accordance with the cost function selected. During program execution, the main program also serves to direct the sequence of execution of the other major sections of the CAMICC algorithm.

Since the plant and compensator are assumed to be linear time-invariant, they may be completely described by transfer functions in the independent Laplace variable  $s$ . Once the plant and assumed form of the series compensation are input along with the profiles forming the desired phase and magnitude response program control proceeds to the minimization subroutine, BOXPLX. From the point of view of computer aided control system design, associated with this subroutine are two key function sub-programs, FE(X) and KE(X). The function sub-program FE(X) performs the calculation of the system open loop frequency response, at the discrete frequencies provided in the input data, and calculates the value of the cost function based upon the difference between the actual and desired frequency response at the discrete frequency values. Six separate cost functions, from which the program user may select one, are incorporated into this function sub-program. The value of the cost function is used by the minimization routine to vary the compensator parameters in order to minimize this quantity. The function sub-program KE(X) checks for any violation of the implicit constraints, which in this particular case are concerned mainly with the presence of right half plane roots. The implicit constraints used, also impose limits on the real and imaginary parts of the numerator and denominator polynomials evaluated at any



Program Block Diagram

Figure IV-1

particular frequency. These latter limits are a direct result of the finite computer word size and the particular method used to represent numerical values within the machine's memory. Both of these factors have an effect in determining the maximum and minimum size numerical value that may be represented within any particular computer. While the coefficient values determined by the minimization routine will not exceed these maximum or minimum values, the real and imaginary parts of the numerator and denominator polynomials evaluated at a particular frequency are squared during the process of determining the magnitude of the frequency response. Without the implicit constraints these squared values can exceed the size of the numerical capability of the machine, particularly in the case of high order systems, causing premature termination of the algorithm and hence failure in obtaining a solution. Once the minimization subroutine determines the coefficients that result in a minimum value of the cost function or the minimization process is terminated for other reasons, program control is transferred back to the main program. Here the roots and gain of the compensator which resulted in a minimum cost function are calculated and output. The open loop frequency response of the system is then calculated using a more densely distributed number of frequency values than would normally be selected by the designer for use in the minimization process. This is done largely to show the existence of any resonant peaks which might otherwise be straddled and not apparent if only the discrete frequencies selected by the designer were used in the plotting routines. The stability of the entire system is then checked by adding the open loop numerator and denominator polynomials and applying the Routh stability criterion to the resulting polynomial which is the system characteristic equation. Finally the results at the discrete frequencies specified by the designer and the "continuous" curve generated by the SIMULA subroutine are output on Bode and gain versus phase

plots. In addition, plots of the difference between the desired and actual results at the discrete frequencies specified by the designer are output in order to aid the designer in making any modifications to the form of the compensator that may be necessary in order to better satisfy the system specifications.

As stated in the previous paragraph six separate cost functions have been incorporated into the function sub-program FE(X). The first three of these are of the error squared type and the latter three are of the absolute error type. These six cost functions represent various combinations of one minus the actual response divided by the desired response at the discrete frequencies under consideration. These pre-defined cost functions are provided as a matter of convenience in using the program and certainly do not prevent the user from defining his own cost function provided this is placed within the FE(X) function sub-program. The first cost function combines the deviation of both the desired magnitude and phase responses from the actual magnitude and phase responses of the open loop system. That is the first cost function is of the form

$$J(\omega) = \sum \left\{ \left( 1.0 - \frac{AMAG(\omega)}{DMAG(\omega)} \right)^2 + \left( 1.0 - \frac{APHASE(\omega)}{DPHASE(\omega)} \right)^2 \right\}$$

where AMAG and APHASE are the actual magnitude and phase response of the open loop system and DMAG and DPHASE are the desired magnitude and phase respectively. These quantities are evaluated in the cost function and summed over the discrete frequency values in the specified range. The second cost function which the user may choose, uses solely the deviation of the actual magnitude response desired magnitude response evaluated at the discrete frequency values. Thus this cost function is of the general form

$$J(\omega) = \sum \left( 1.0 - \frac{AMAG(\omega)}{DMAG(\omega)} \right)^2$$

If a minimum phase system is being considered the magnitude and phase plots as a function of frequency are uniquely related by the Bode theorems [3]. That is, determining the magnitude ratio of a system over a given frequency range will also result in a unique phase relationship for the system. Thus if the system design being considered is of the minimum phase type the use of the type two cost function described should result in both the magnitude and phase specifications being satisfied. The third pre-defined cost function uses the deviation of the actual open loop phase response from the desired phase response in a function of the form

$$J(\omega) = \sum \left( 1.0 - \frac{APhase(\omega)}{DPhase(\omega)} \right)^2$$

While for minimum phase systems the magnitude and phase responses are uniquely related, it should be noted that the fixed loss or gain of the system cannot be determined from the phase characteristics alone. Consequently the use of the type three cost function may result in a constant gain error for the magnitude response of the system. This error however is readily observable from the output produced by the program and correction of this may be accomplished by adjustment of the compensator gain value accordingly. The other three pre-defined cost functions are of the same general form as those just presented with the exception that the squared differences are replaced by the absolute value of the differences. The general rationale in selecting this latter type of cost function was that on occasion the squared type cost function will have a tendency to flatten near the minimum. With the absolute value cost function the user may avoid the excessive time required in finding the



minimum values when the case arises where the squared cost function becomes flat.

In utilizing this design program the user should also be aware that the need for frequency scaling of the problem may arise. This is particularly true when high order systems involving high frequencies are being considered. The need for frequency scaling is primarily a result of the finite word size available in the computer system. It must be kept in mind that the magnitude response of the system is computed as the ratio of two complex numbers and while the resulting magnitude may be relatively small the value of the magnitudes of the individual complex numbers forming this ratio may be large. The situation is further complicated by the fact that in computing the magnitudes of the complex numbers involved the real and imaginary parts must be squared, thus effectively reducing the maximum power of the numbers which can be allowed by a factor of two. If the magnitudes of the resulting numbers become too large an error message will be printed recommending that the problem be frequency scaled.

## B. DATA INPUT AND OUTPUT

In order to simplify use of the program, an attempt has been made to maintain the input information necessary for execution of the algorithm to a minimum, and at the same time preserve a certain amount of flexibility within the program which may be controlled by the user. In so doing some of the program control variables for various options have been set internal to the program and may not be manipulated by the user in the input data deck. These internally set variables do not detract from the general functioning of the program, but mention of this is made to

bring it to the user's attention in the event that specialized problems might more effectively be investigated by resetting some of these normally preset program variables. Many of the preset items are concerned with control of the minimization routine. Such items as the upper and lower bounds of the search area, the integer programming option, and the seed for the random number generator are set to specific values within the main program just prior to calling the minimization subroutine. For example for the minimum phase case, the lower and upper bounds on the search areas of the compensator numerator and denominator polynomial coefficients are set at 0.0 and  $1.6 \times 10^6$  respectively. For the non-minimum phase case the bounds on the search areas of the numerator polynomial coefficients only are extended to  $-1.6 \times 10^6$  and  $1.6 \times 10^6$  and the implicit constraints are adjusted to allow right half plane roots for the numerator polynomial. The integer programming option for BOXPLX is also preset to a value of zero indicating that values for the polynomial coefficients need not be integers. In general, selection of the integer programming option results in considerably more execution time for the program and for the purpose of varying the compensator parameters the selection of this option provided nothing to enhance the solution of problems.

The input data required for program execution and control may be divided into general areas as follows: 1.) a title card; 2.) a series of cards giving the gain and coefficients of the plant transfer function; 3.) a series of cards giving the assumed gain and coefficient values of the compensator transfer function; 4.) a control card describing the plots desired, the type of cost function to be used, whether the system contains a zero order hold, and the number and range of the discrete frequency points being



considered; 5.) cards containing the discrete frequency values, the desired magnitude, and the desired phase; 6.) finally a card describing the number of iterative trials to be allowed in the minimization and whether any output is desired from the minimization routine in order to keep track of the manner in which the coefficients are being varied. The specific input formats of the above necessary data are described in detail in the following paragraphs. These paragraphs are numbered so as to indicate into which of the six general areas a particular data card belongs. This description coupled with the program listing given in Appendix E should provide sufficient information for setting up the data deck necessary for execution of the program. Figure IV-2 illustrates the data deck used in executing the first example problem presented in section C of this chapter. In this figure each line of data represents an individual data card. The column numbers for the data cards are also shown above and below the data deck section of the figure.

#### 1. Title card.

This must be the first card of the data deck and provides a means of problem identification if several separate problems are run under one job name. Columns 1 through 48 may be used to input whatever alpha-numeric characters the user desires for identifying any particular problem. This title will also be printed on the graphical output requested by the user.

#### 2A. Plant gain

The second card contains the gain of the plant in columns 1 through 10. If the gain is not explicitly present in the plant transfer function then the value input on this

card should be unity. The format used to read the plant gain is an E10.0 type format. Thus, the gain may be input in either floating point or scientific notation type formats.

2E. Input form and order of the plant transfer function numerator.

This card specifies whether the numerator of the plant transfer function is to be read in factored or polynomial form and what the order of the numerator is. A P or an F in column 1 indicates whether the plant numerator data is to be read in polynomial or factored form respectively. Columns 2 and 3 contain the order of the numerator. Thus an input of the form FC3 starting in column 1 indicates that the polynomial is of third order and is to be read in factored form.

2C. Plant transfer function numerator.

The following card(s) contain either the polynomial coefficients or the factors of the plant transfer function numerator, depending on which form has been chosen for input. If the polynomial form of input has been selected then the polynomial coefficients are input in ascending powers of  $s$  using an E10.0 format. If more than eight coefficients are needed they are simply continued on successive cards. The coefficient of the highest order term is assumed to be normalized to unity and is not read in explicit. Thus the number of coefficients actually read corresponds to the order of the numerator polynomial. If the factored form of input is chosen then each factor is read in on a separate card with columns 1 through 10 being used for the real part of the factor and columns 11 through 20 for the imaginary part. Again the format used is E10.0,

thus allowing the factors to be input in either floating point or scientific notation form. Since complex factors will occur in conjugate pairs the values of these factors need only be input once. Thus the number of cards used will vary depending upon how many purely real and how many complex factors are involved.

#### 2E. Input form and order of the plant transfer function denominator.

Again the form of input is specified as in card number 2E along with the order of the plant transfer function denominator.

#### 2E. Plant transfer function denominator.

The format used to input the denominator is identical to that used for input of the numerator, contingent upon the desired form specified for the input. That is to say, it is possible to input the numerator in polynomial form and the denominator in factored form, or vice versa, since each time a polynomial is input as data the input form is uniquely specified.

#### 3A. Assumed compensator transfer function gain.

Following the input of the information about the plant the description of the form of the compensator must be supplied. This is begun by inputting an assumed compensator gain using the same format as was used to specify the plant gain. Due to the manner in which the compensator transfer function numerator coefficients are handled within the program, if the assumed compensator gain is set to zero, for lack of any better value, this will result in the starting

guess of all the numerator coefficients being zero. This will not cause the program to terminate prematurely, but the initial guess of zero for all the coefficients may not be what is desired by the user. If doubt exists as to what value gain should be assumed, a value of 1 is suggested as a convenient arbitrary starting point.

### 3B. Input form and order of the assumed compensator numerator.

The method for specifying the form of the input for the compensator initial numerator parameters is identical as that used in specifying the plant numerator. A P indicates polynomial form and an F indicates factored form.

### 3C. Assumed initial values of compensator numerator parameters.

In order to maintain continuity in the input format the scheme and format used to input the compensator numerator parameters is identical to that used in specifying the plant transfer function numerator.

### 3D. Input form and order of the assumed compensator denominator.

As in previous cases either polynomial or factored form is specified along with the order of the compensator denominator.

### 3E. Assumed initial values of compensator denominator parameters.

Once again the same format as used to input the plant

data is used in order to maintain the continuity of the input formatting.

#### 4A. Program control card.

The program control card is used to input various quantities controlling both the range and number of discrete points used in the minimization of the cost function, the output desired from the program, and also whether a zero order hold is present in the error channel of the system. As many as twelve separate quantities may be input on this single control card in order to direct execution of the program. Columns 1 through 10 specify the minimum frequency ( $\omega_{MIN}$ ), in radians, of the total range being considered. Columns 11 through 20 specify the maximum value ( $\omega_{MAX}$ ), in radians, of the range of frequencies being considered. Both of these values are read using a 2E10.0 format. Following the maximum and minimum frequency values, up to ten input quantities are read using a 10I3 format. Columns 21 through 23 specify the number of discrete frequency points (NOMEG) over the given range at which the open loop frequency response is to be determined and the cost function evaluated. The next input quantity (KNOW), in columns 24 through 26 specifies whether the discrete frequencies of interest are to be read from data cards or automatically computed. Normally this quantity is set to unity indicating that the discrete frequency points of interest will be read from data cards. A value of zero indicates that the discrete frequency values are to be incremented linearly by the program starting at the minimum frequency value. A value of two indicates that the discrete frequency points will be incremented logarithmically beginning with the minimum frequency value. Columns 27 through 29 indicate whether a Bode plot of the results is desired. If this variable (NECDF) is set equal to one no Bode plot will be

output. A value of zero for NBODE will result in both magnitude and phase plots. Columns 30 through 32 control the output of the Nyquist plot. If NYQST is set to unity then the Nyquist plot will be suppressed. A value of zero for NYQST will result in a Nyquist plot output, however this is plotted only over the range of frequencies between WMIN and WMAX. Columns 33 through 35 indicate the presence of a zero order hold (IZOH) in the error channel of the unity feedback system. With IZOH set to zero the system is treated as being of a continuous nature. With IZOE set equal to one a zero order hold is considered to be present in the error channel and the open loop magnitude and phase computations are modified accordingly. Columns 36 through 38 (NICHCI) specify whether a magnitude versus phase plot of the results are desired. As with the other plot options, a value of zero will result in a plot being produced while a value of one will suppress the magnitude versus phase plot as part of the output. Columns 39 through 41 contain the variable indicating whether non-minimum phase solutions are to be considered. With this variable (NMNFS) set equal to zero only minimum phase solutions will be allowed. A value of one indicates that a non-minimum phase solution will be allowed provided that it is the one which minimizes the cost function. Columns 42 through 44 indicate if the desired magnitude profile values to be input are in decibels. If this variable (IDB) is zero the magnitude values to be read are dimensionless gain values, while a value of one indicates that the magnitude values to be input will be in decibels. Columns 45 through 47 contain the variable IPLOT. This quantity is used to choose between printer plots or Calcomp plots for the graphical output. A value of zero will result in printer plots and a value of unity will provide Calcomp output of the graphs desired. It is recommended that the printer plot option be used in initially solving a particular problem since this type of output requires less turn around time than the Calcomp



output. Finally columns 48 through 50 specify the type of cost function (ICOST) the user wishes to choose among the six preprogrammed functions available in subroutine FE(X). The value specified should be between 1 and 6 corresponding to the particular cost function desired. If this value is zero the program will default to the type one cost function.

#### 4E. Sampling period.

This card must be inserted only if a zero order hold is specified in the error signal channel. The sampling period  $T$ , in seconds, is specified in columns 1 through 10. If the system being considered is of the continuous type then this card must be omitted.

#### 5A. Discrete frequency values considered.

Next follows a set of data cards specifying the values in radians per second of the frequencies at which the desired gain and phase values of the open loop frequency response are specified and at which the cost function is to be calculated. These values are input in an 8E10.0 format using as many cards as necessary to specify all the frequency values at which the program is to execute the algorithm.

#### 5E. Gain profile of the desired open loop response.

This set of data cards contains the desired open loop gain response at the discrete frequency values previously specified. The same 8E10.0 format is used here as was the case in reading the discrete frequency values. The same number of cards should be used as was used in specifying the



discrete frequency values.

#### 5C. Phase profile of the desired response.

Here follows a series of cards specifying the desired open loop phase response, in degrees, at the discrete frequency values being considered.

#### 6. Minimization trials card.

Here both the number of trials to be allowed in the minimization process and the print interval for diagnostic purposes are specified. These quantities are input using a 2I5 format. If the number of trials is zero then a default value of 2000 is assumed. If the print interval is specified as zero then no output from the minimization routine will result. Caution should be exercised in specifying the print interval in that if this interval is made too small an excessive amount of printed output may result. The program user is referred to the comment section of subroutine BOXPLX in Appendix A for guidelines in selection of the print interval.

[illegible]

COMF. CFT. EXMP. ONE LEAD COMP. 10PTS

[illegible]

12345678901234567890123456789012345678901234567890  
COLUMN NUMBERS, READ VERTICALLY)

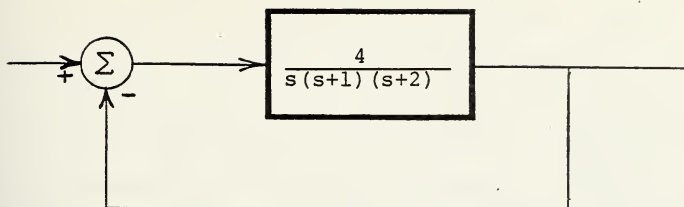
48

## C. SERIES COMPENSATOR DESIGN EXAMPLE PROBLEMS

In this section a number of example problems are presented and discussed in order to illustrate the use of the computer aided compensator design algorithm. In the hope of presenting a clear, concise illustration of the algorithm and its use the example problems begin with relatively simple straight forward textbook examples and gradually progress to problems of a slightly more difficult nature. Also included are problems that are intended to illustrate some limitations of the program in that these problems were not completely solved by the algorithm or they may have required a certain amount of manipulation of the input data in order to obtain a solution. These latter types of problems have been included since it is important that the designer using any computer aided design algorithm understand the algorithm's limitations and peculiarities as well as its intended capabilities. It is not intended to mislead the reader by implying that the example problems presented here provide a comprehensive presentation of all possible limitations of this automated design program. Rather they represent those limitations which were encountered in the process of verifying the program over a necessarily limited period of time. With this in mind the first example problem which deals with a relatively simple lead compensator is presented.

### 1. Compensator Design, Example 1.

Consider the uncompensated system shown in figure IV-3.



Design Example 1, Block Diagram

Figure IV-3

The asymptotic Bode diagram for the open loop system is shown in figure IV-3A. As can be seen from the open loop Bode plot, even though the system is stable it is lightly damped. A single section lead compensator is to be used to increase the phase margin. A complete discussion of the reasons for selection of the lead compensator and a detailed discussion of this particular example is presented by Thaler and Brown [18]. The transfer function form of the single section lead compensator consists of a first order numerator and a first order denominator. A desired magnitude and phase profile, consisting of ten discrete frequency points over the range from 0.2 to 20.0 radians, was selected to give a phase margin of approximately 45 degrees in order to increase the damping of the system. As an initial guess the compensator was assumed to have a gain of 1.0, a zero at the origin and a pole at -1.0. This information was supplied to the program in the prescribed format and the optimized results in the form of magnitude and phase plots are shown in figure IV-3E. Here, as throughout the remainder of the example problems, the diamond shaped symbols represent the desired values specified by the program user, the X symbols represent the values computed by the program at the discrete

frequency values specified using the compensator parameters returned as the solution from the minimization routine, and the continuous curve represents the response computed by the program again using the parameters returned as a solution but with a more dense set of frequency values used in order to show any resonant peaks or anomalies in the magnitude and phase curves which might be difficult to visualize from a sparsely selected set of discrete values. As can be seen in figure IV-3E the results match the desired response well over the range of interest. Figure IV-3C shows the same results using the magnitude versus phase plot to portray the open loop frequency response. The numerical values of the compensator parameters returned from the program as a solution along with the starting values used are shown in figures IV-3D, IV-3E, and IV-3F. A comparison with the solution values given in ref. 18 will show that they agree. Figure IV-3G shows plots of the differences between the actual and desired open loop frequency response of the resulting system at the discrete frequencies considered in the minimization process.

This problem as described required approximately 500 iterations within the minimization routine to achieve the solution shown. This however includes two restarts to attempt insuring that a global minimum had been achieved and that termination was not due to a local minimum. From the beginning to the first restart required approximately 250 trials with the cost function being reduced from a value of 6.33 to a value of 0.00711 (recall that the minimum achievable under ideal condition is zero which due to machine errors alone in representing numbers would not likely ever be achieved). The entire solution required 32.5 seconds of CPU time, a good 10 seconds of which was used in the production of Calcomp plots. These numbers concerning the time required for solution are presented for what they are worth in that extrapolation of these numbers to more

complicated problems is not a straight forward task. As mentioned in Chapt III the minimization routine does become inefficient when the number of variable parameters is allowed to become large. It was felt that a detailed investigation of the efficiency and time required for solution of problems as a function of the number of variable parameters and the number of discrete frequency values to be considered was beyond the scope of the basic development of the program. However, the amount of time required for solution of some of the example problems that follow will be given in order to give the user an "intuitive feel" for the approximate amount of time that may be need to handle certain types of problems.

In order to illustrate that the starting point chosen should have reasonable values, but that the exact values chosen should not effect the final results significantly, the same problem was run a second time with the pole and zero of the compensator both chosen to be at the origin. Again an assumed initial gain of one was chosen for the compensator. The graphical results of the solution are shown in figures IV-3H through IV-3J. The numerical results returned using these new starting values are not shown here since they were identical to within three decimal places with those values shown previously.

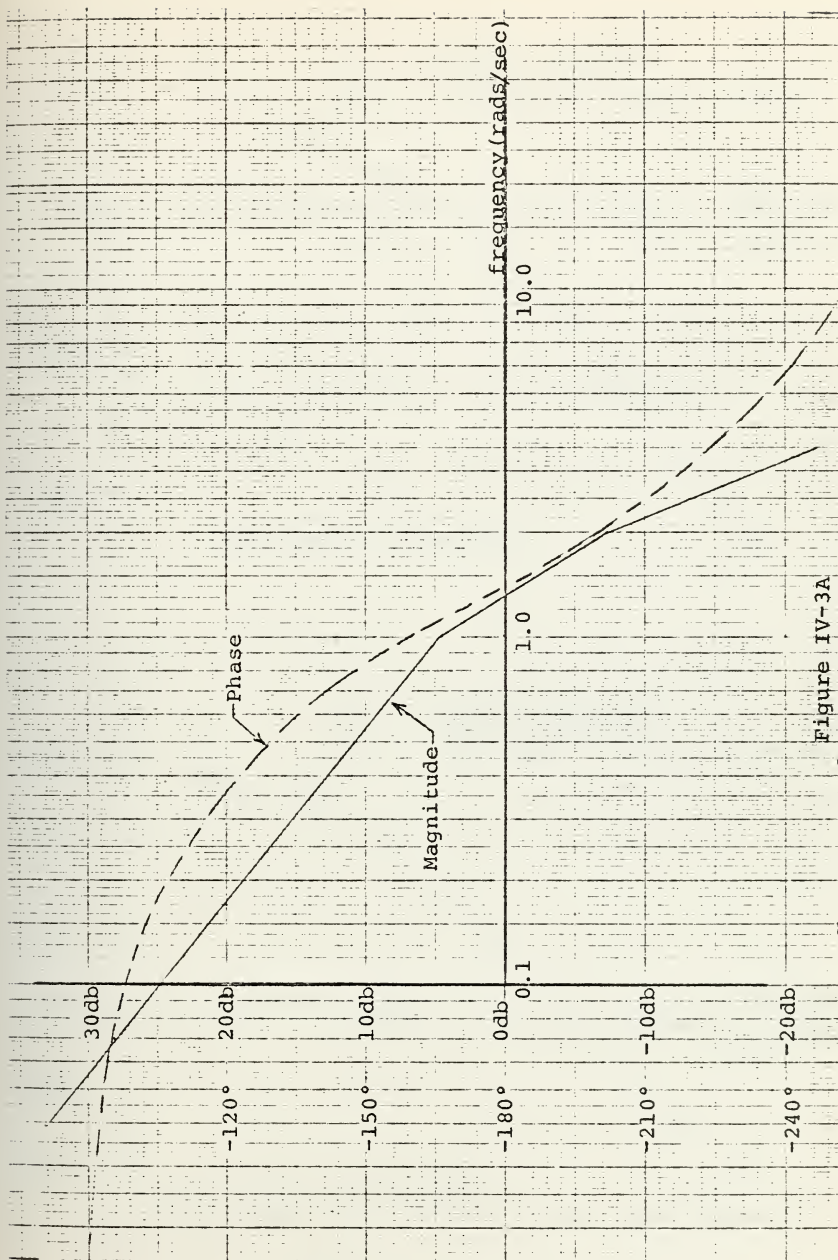


Figure IV-3A  
Uncompensated Open Loop System Bode Plot



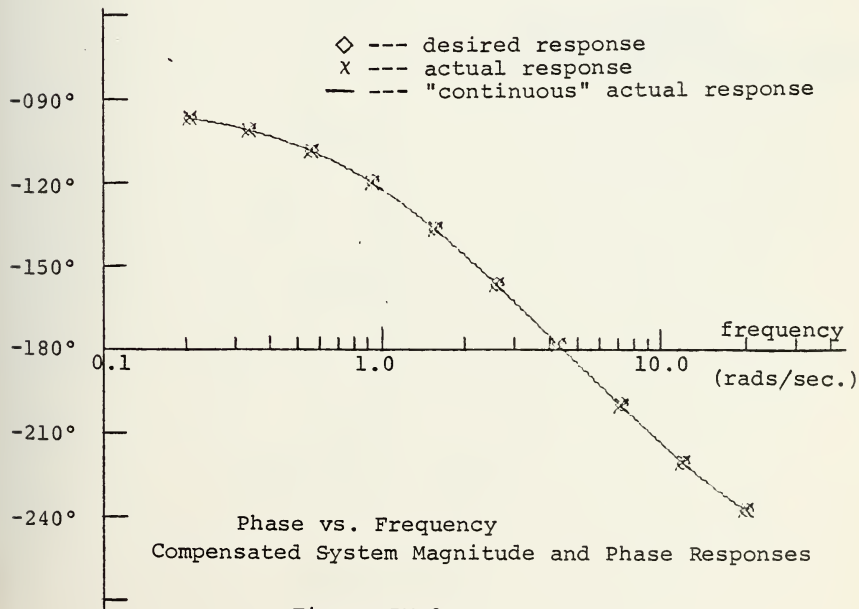
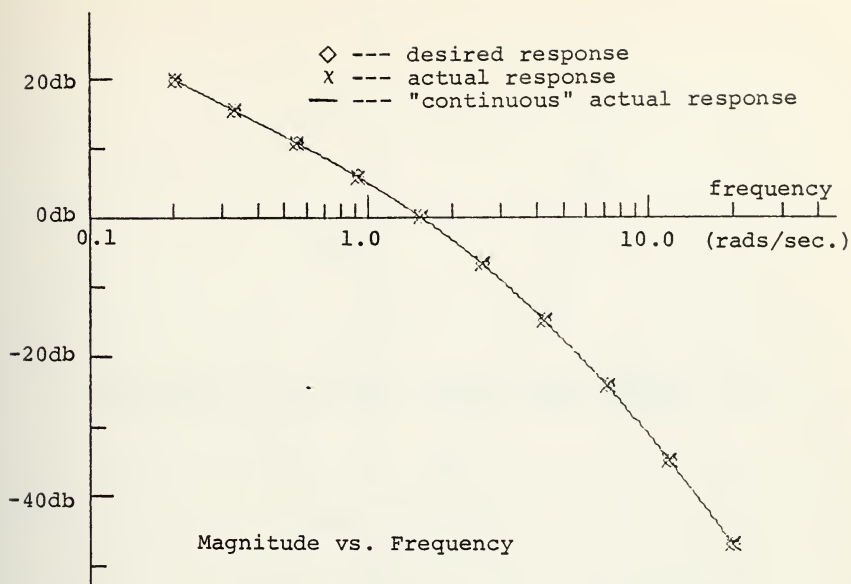
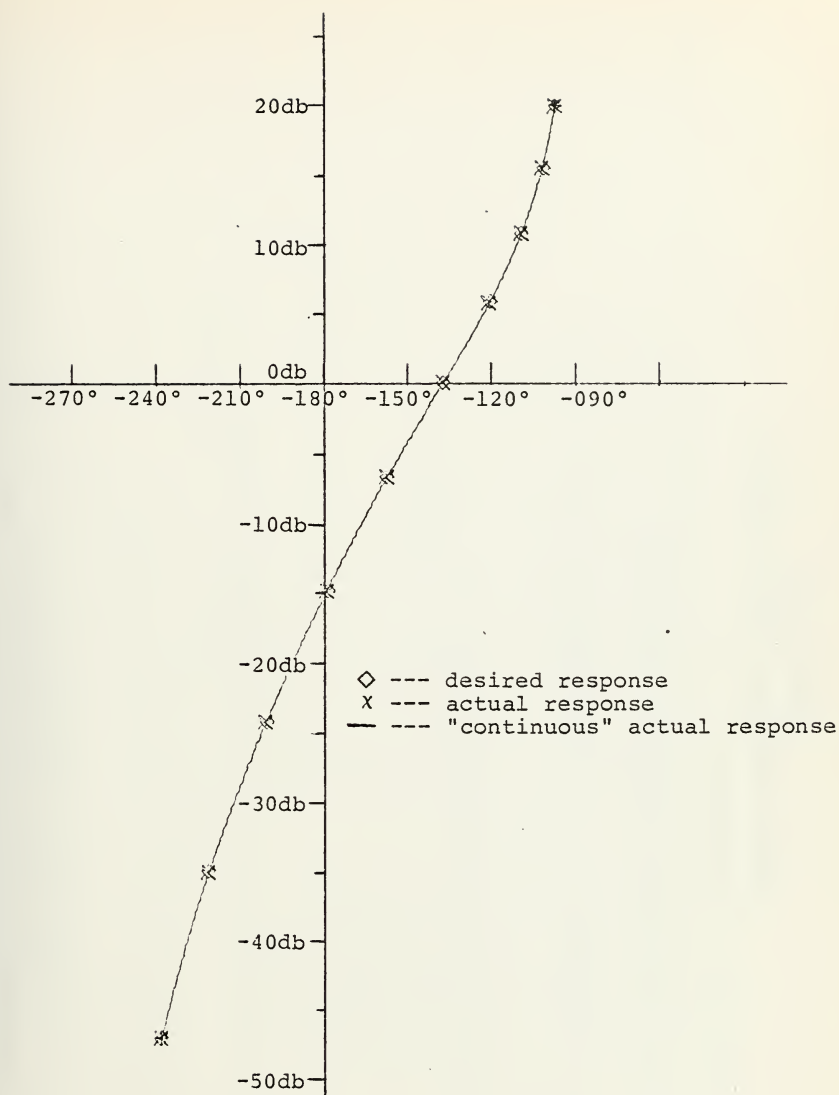


Figure IV-3B



Magnitude vs. Phase

Figure IV-3C

TITLE --- COMP. OPT. EX. CCL LAD COMP LOPTS

UNCOMPENSATED TRANSFER FUNCTION GAIN = 4.000000E 00

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

0.0 2.000000E 00 3.000000E 00 1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION COMPLEX ROOTS  
ARE: REAL PART IMAGINARY PART

0.0 0.0

-1.000000E 00 0.0

-2.000000E 00 0.0

COMPENSATOR TRANSFER FUNCTION GAIN = 1.000000E 00  
COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

0.0 1.000000E 00

COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ARE: REAL PART IMAGINARY PART

0.0 0.0

COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 1.000000E 00

Computer Numerical Output Example 1

Figure IV-3D

COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS  
REAL PART

-1.00000E-00 0.0

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE  
TYPE, THEREFORE NO ROOT PAIR HAVING ZERO VALUE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

THE TOTAL NUMBER OF TRIALS CALLED FOR = 5000

THE COST FUNCTION TO BE USED IS THE TYPE 1

THE MINIMUM COST FUNCTION VALUE = 7.010020E-05

THE ERROR RETURN CODE FFEFBXPLX = 0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

9.999060E 05 9.994820E 05

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE:

REAL PART IMAGINARY PART  
-1.001420E 00 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

9.999400E 05 9.993000E 04

Computer Numerical Output Example 1

Figure IV-3E

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE: REAL PART IMAGINARY PART

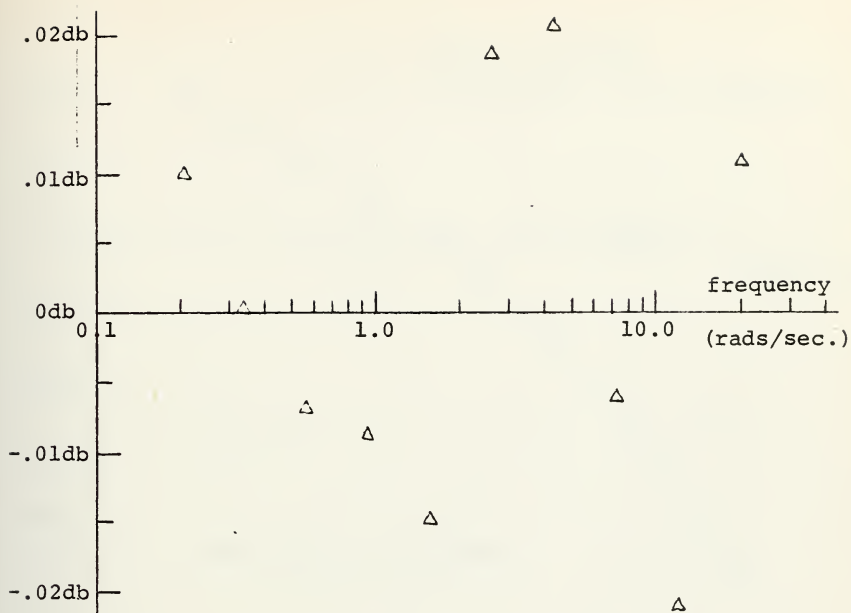
-1.001554E 01 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 1.001596E 01

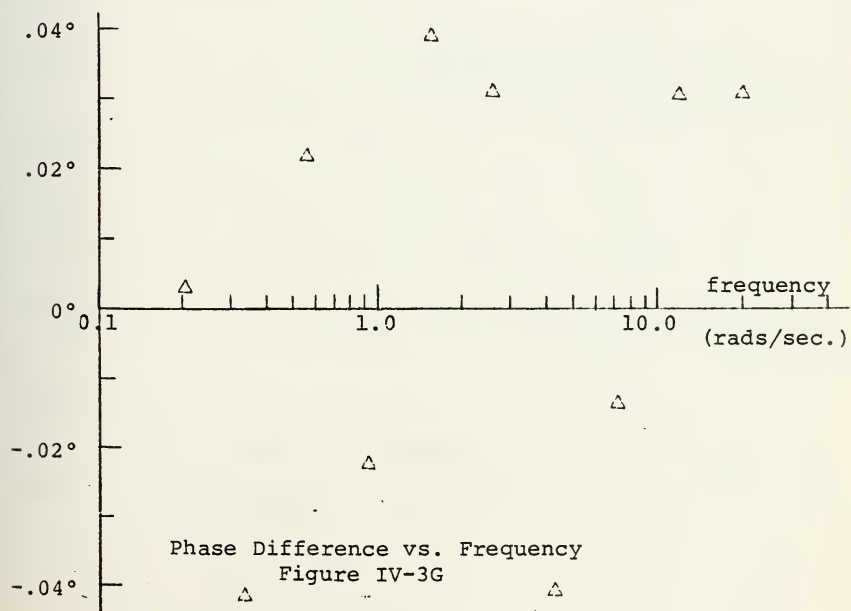
FREQUENCY	MAGNITUDE (DB)	DESIRE D MAG (DB)	PHASE (DEG)	DESIRE D PHASE
2.000000E 01	-1.556243	1.956643	-9.687019	-9.689999
3.340000E 01	1.342311	1.941759	-1.014153	-1.010000
5.570000E 01	1.077661	1.076152	-1.087891	-1.090000
9.280000E 01	5.751251	5.802528	-1.292251	-1.290000
1.544999E 00	3.141840	3.643113	-1.360937	-1.370000
2.580000E 00	-6.743149	-6.743149	-1.563839	-1.570000
4.305555E 00	-1.491242	-1.433965	-1.789066	-1.780000
7.190000E 00	-2.436720	-2.436720	-2.001377	-2.000000
1.200000E 01	-3.511035	-3.503974	-2.266923	-2.210000
2.000000E 01	-4.762190	-4.793379	-2.373905	-2.380000

THE ROOTS TEST OF THE CHARACTERISTIC EQUATION INDICATES  
THAT THE SYSTEM IS STABLE

Computer Numerical Output Example 1  
Figure IV-3F



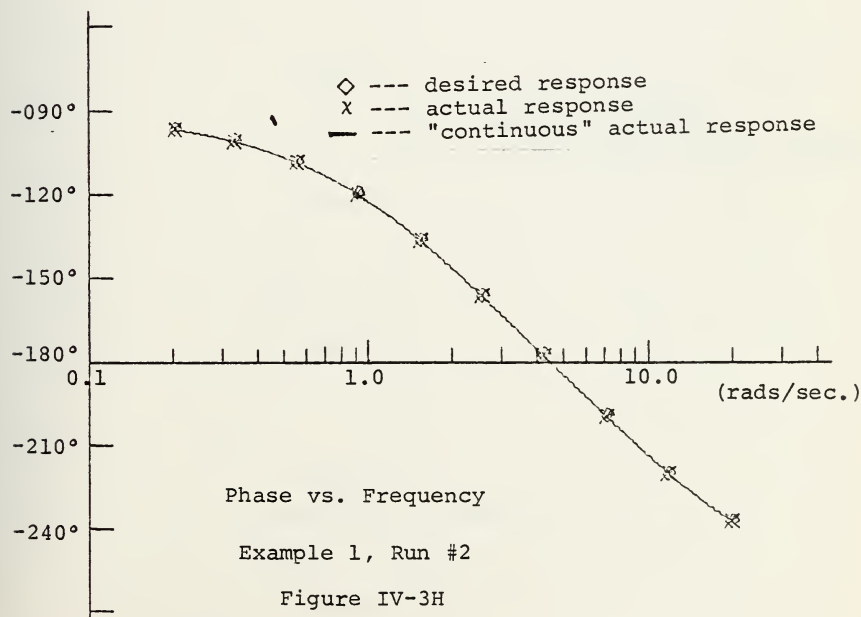
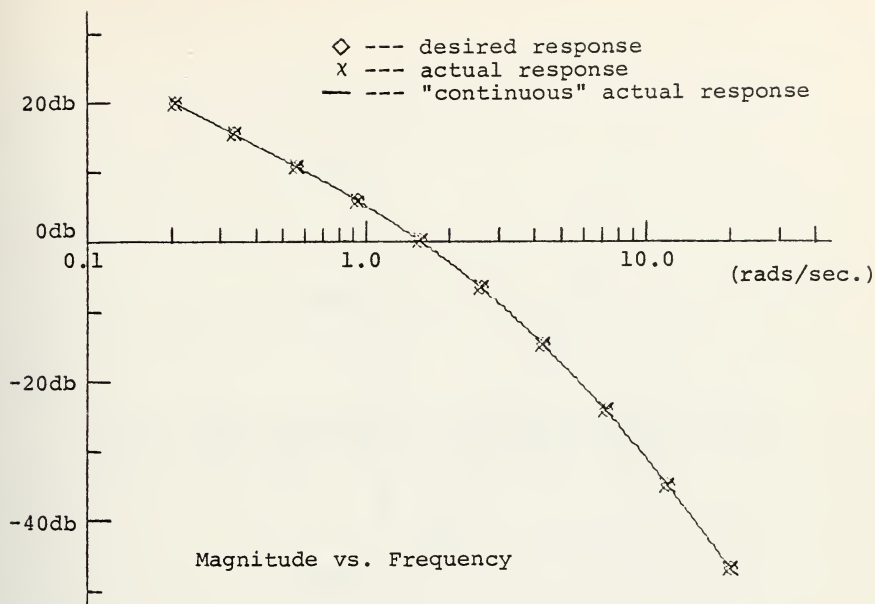
Magnitude Difference vs. Frequency

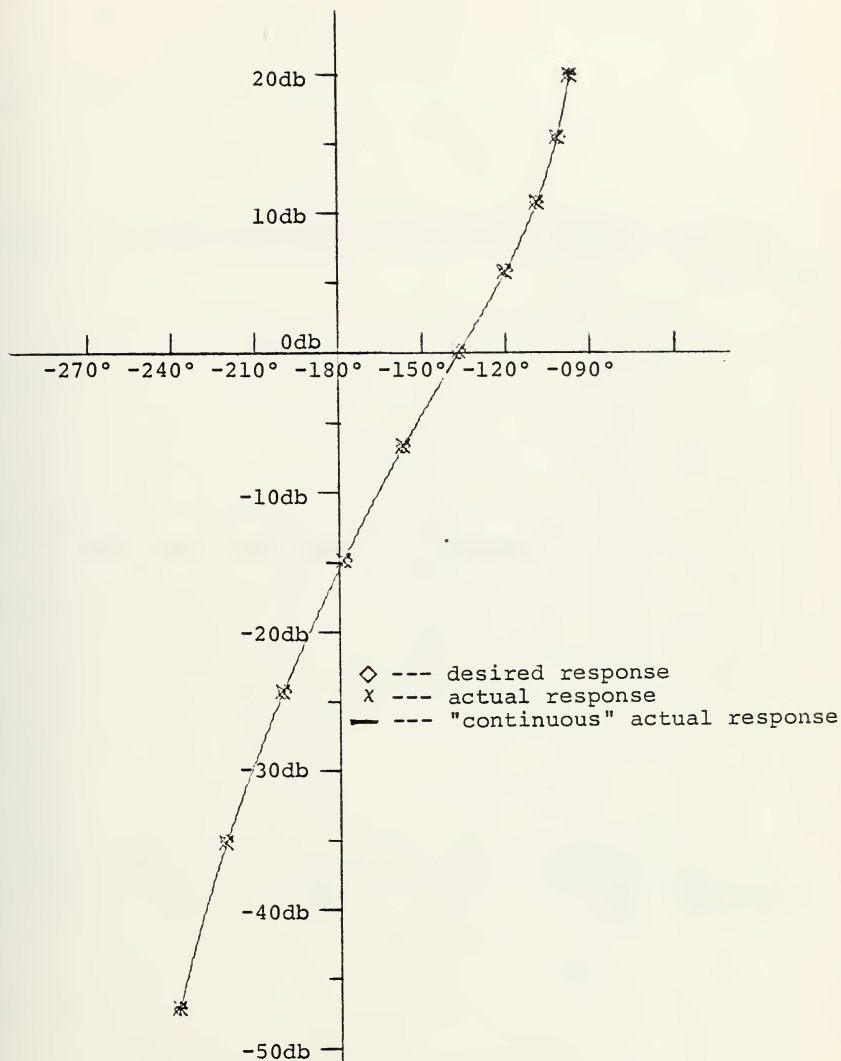


Phase Difference vs. Frequency

Figure IV-3G



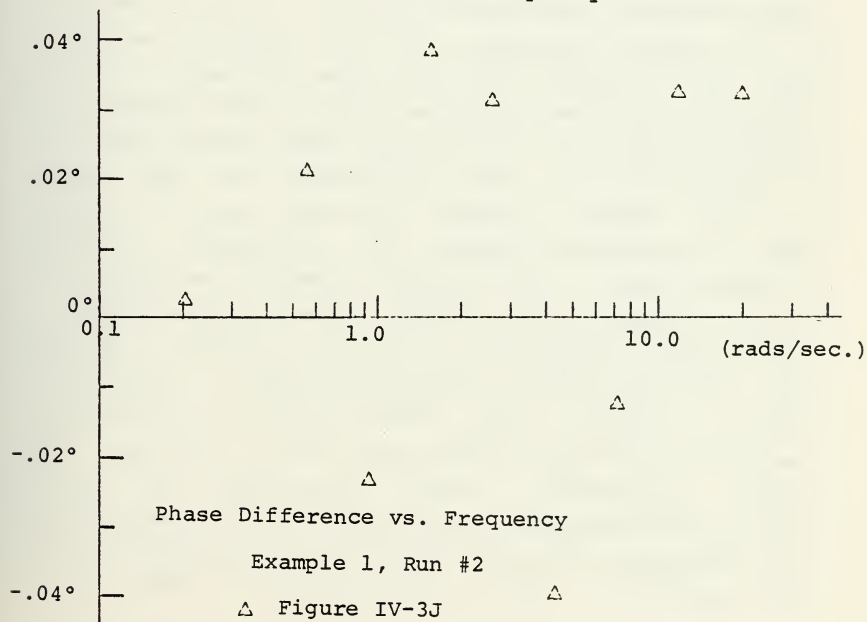
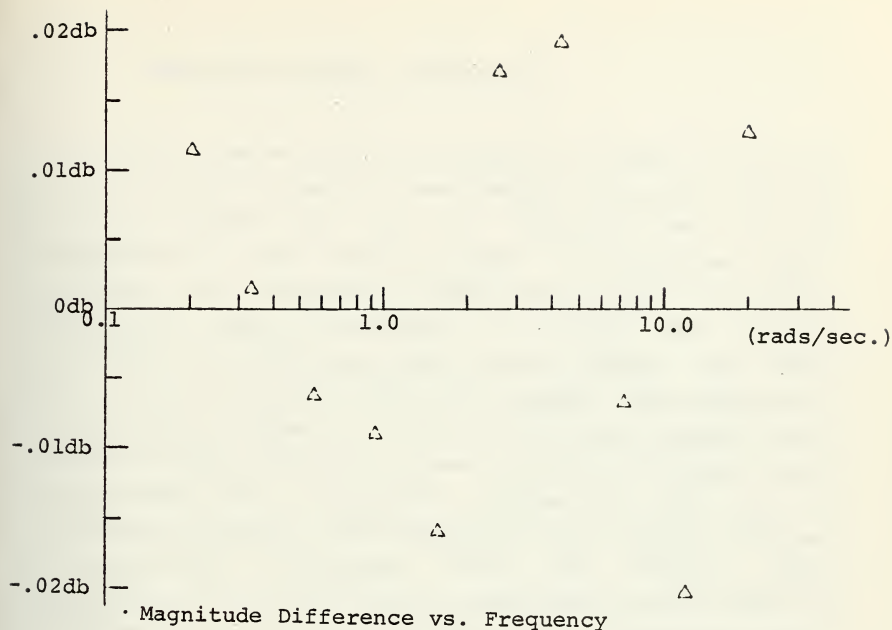




Magnitude vs. Phase

Example 1, Run#2

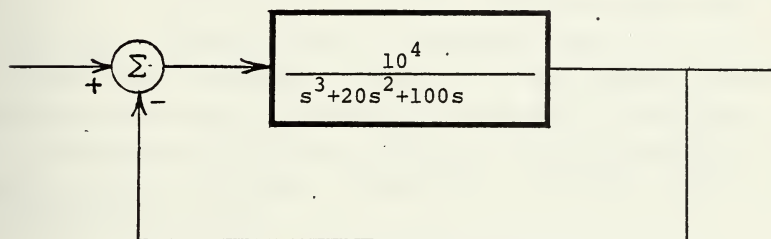
Figure IV-3I



## 2. Compensator Design, Example 2.

This example problem (discussed in detail in ref.10) has been chosen not only to familiarize the reader with the use of the program, but also to illustrate one of the shortcomings of the design program. In particular, while the program was successful in finding the correct solution, difficulties were encountered in the root finding subroutines. In this particular problem, during the final phases of execution the roots of a fourth order polynomial must be found in order to calculate the phase response using subroutine SIMULA which does these calculations using a more densely spaced set of discrete frequencies than in the minimization routine. Since numerical root finding routines have difficulty in finding roots for fourth order polynomials, the lack of good convergence for the roots of the system in both root finding subroutines caused the phase calculations normally done by subroutine SIMULA to be aborted. It should be emphasized that failure of the root finding subroutines in this case only effects the generation of the "continuous" phase response and in no way created any problems for the minimization part of the program as evidenced by the resulting correct values for the compensator parameters. The original uncompensated system being considered is shown in figure IV-4. A Bode diagram of the uncompensated system (figure IV-4A) reveals that the system as it exists is unstable. If stability were the only consideration, a simple attenuation of the plant gain would be sufficient to provide adequate phase margin to insure this. However, as with many series compensation problems, here we must effectively strike a compromise between effectiveness of control (generally associated with a higher gain) and stability of the system (generally associated with a lower gain). As discussed in ref. 10 a simple gain

attenuation would result in stability being achieved, but also has an undesirable effect upon the velocity error constant. Thus as discussed in detail in ref. 10, a lag compensation network appears as the best candidate to accomplish the job.



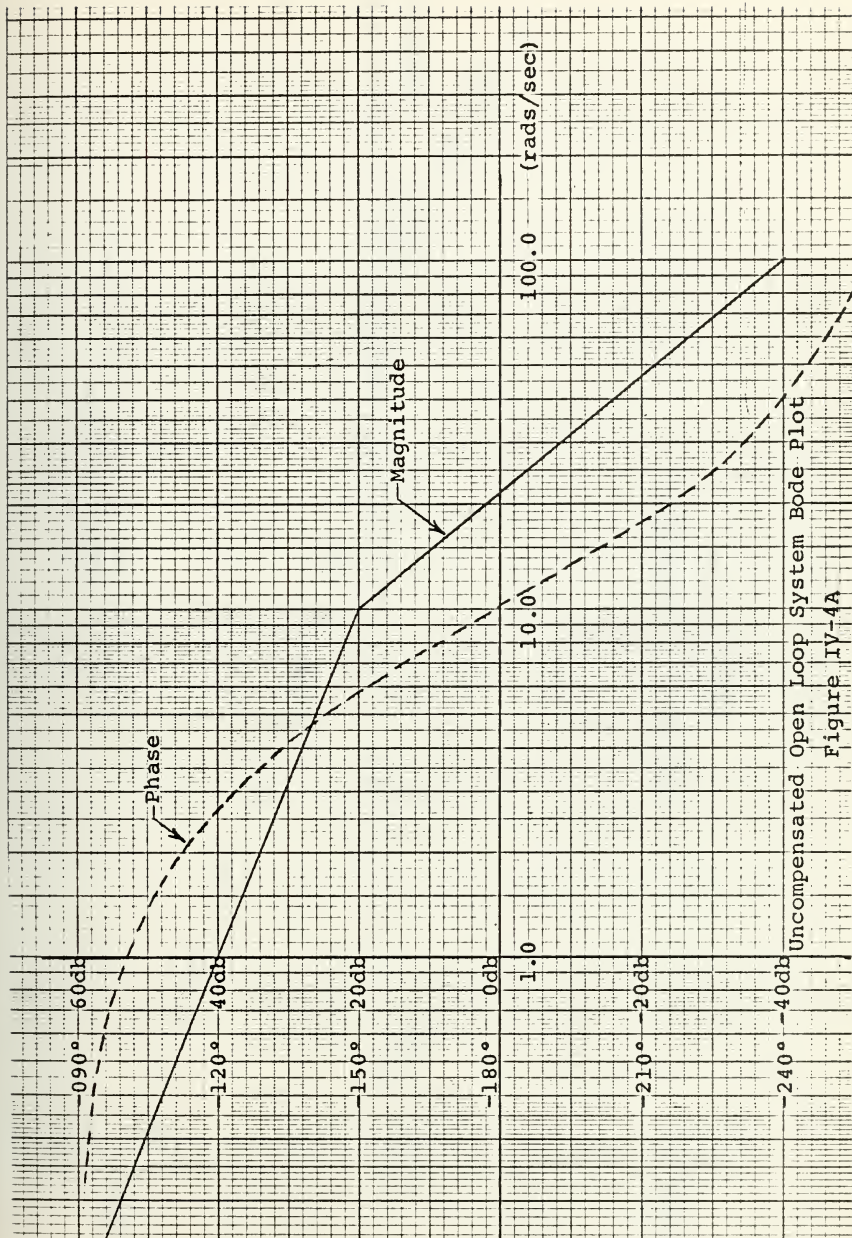
Design Example 2, Block Diagram

Figure IV-4

In this particular problem twenty discrete frequency values over the range from 0.005 to 50.0 radians were selected and the desired magnitude and phase values specified at these frequencies. As in the previous example problem the form of the compensator is selected as having a first order numerator and first order denominator, that is in this case a single section lag compensator is to be used. A type one form of the cost function was selected for the minimization process. The magnitude and phase profiles of the desired response along with the resulting response with the compensator included in the open loop system are shown in figures IV-4B, IV-4C, and IV-4D. As mentioned, the continuous curve for the phase response was not computed because of the failure of the root finding routines to find the open loop system roots. It can be seen in figure IV-4C that the phase response does agree well at the discrete

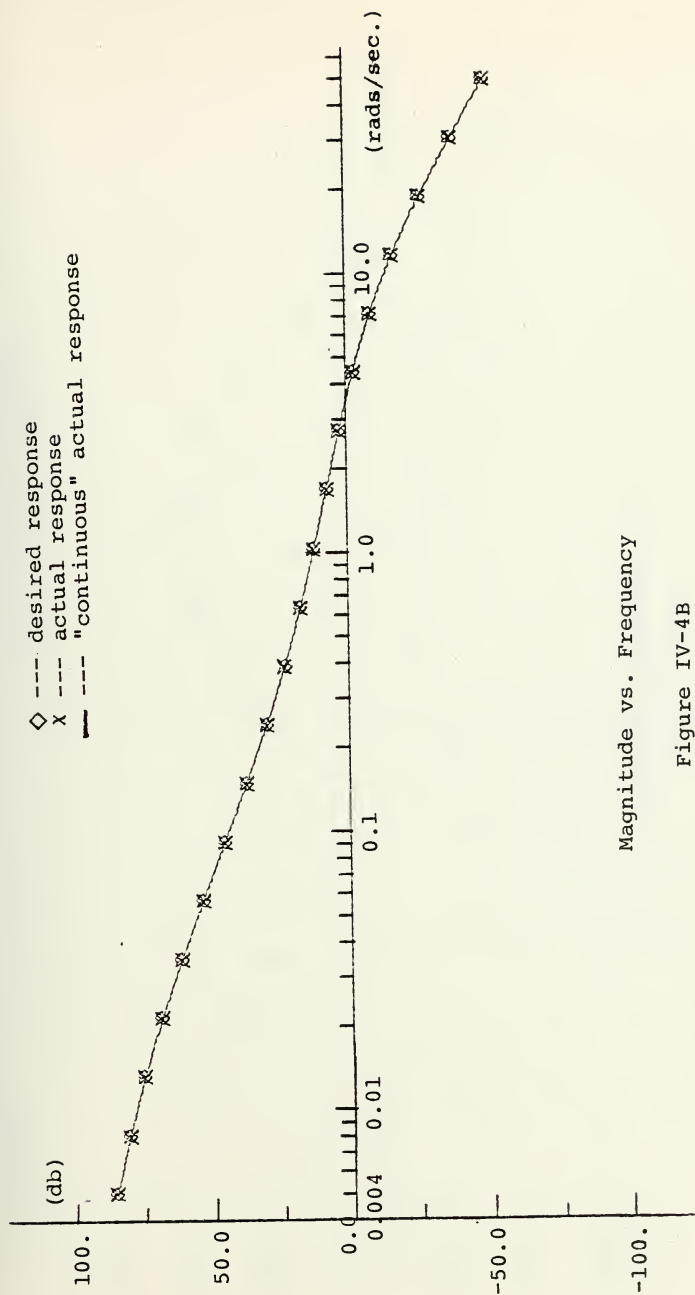
frequencies specified. Since we are dealing with a minimum phase system and figure IV-4B shows that there exists no resonant peaks in the magnitude curve the user may conclude that the phase response will also be well behaved for the various frequency values between the explicitly specified discrete values plotted. This case should serve to illustrate that the program requires that the user still provide an interpretation of the results. The computer output of figures IV-4E, IV-4F, and IV-4G shows that the compensator parameters are in agreement with the problem solution as presented in ref. 10 if the reader wishes to make the comparison. Figures IV-4H and IV-4I show plots of the differences between the actual and desired magnitude and phase at the discrete frequencies selected for the minimization process.





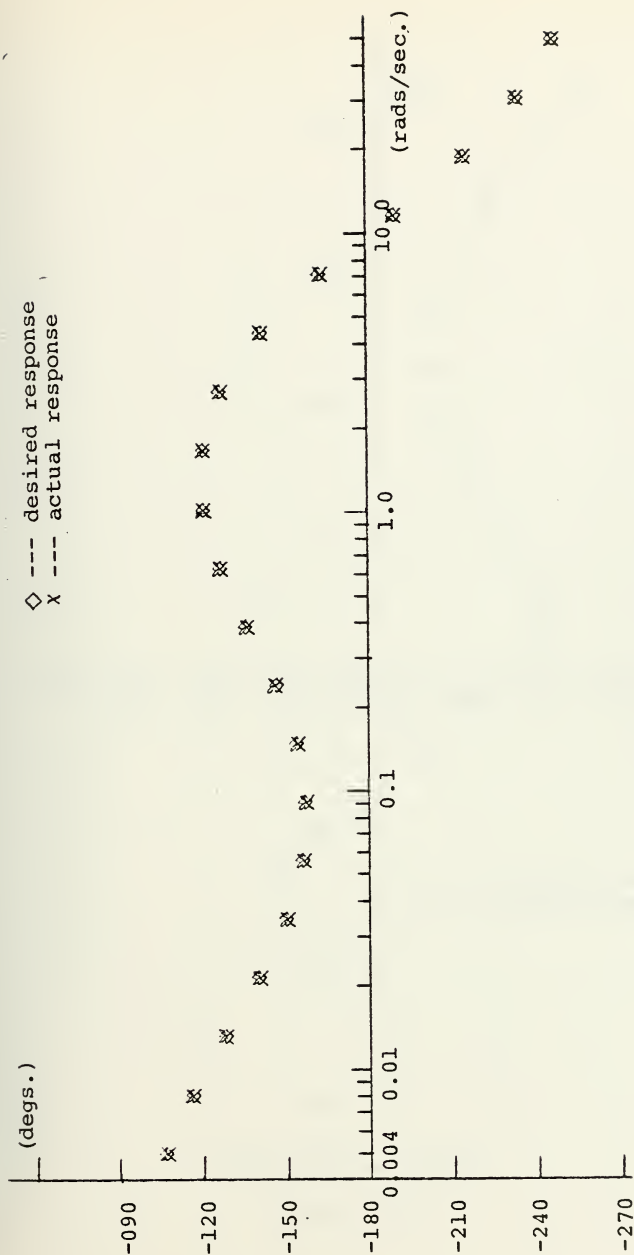
Uncompensated Open Loop System Bode Plot

Figure IV-4A



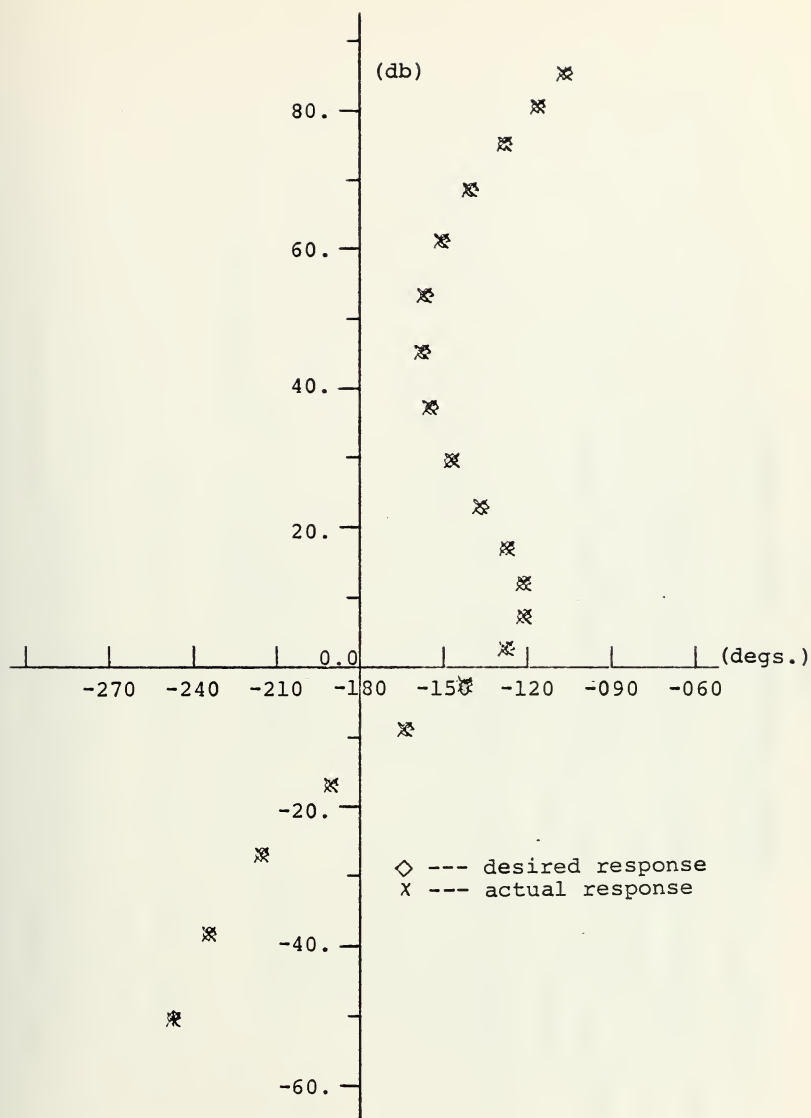
Magnitude vs. Frequency

Figure IV-4B



Phase vs. Frequency

Figure IV-4C



Magnitude vs. Phase

Figure IV-4D

TITLE --- COMP. FTIM17. EXAMPLE 2 LAG COMP. 20PT

UNCOMPENSATED TRANSFER FUNCTION GAIN = 1.000000E-04

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E-00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

0.0 1.000000E-02 4.000000E-01 1.000000E-00

ARE: UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS  
REAL PART IMAGINARY PART

0.0 0.0

-1.000000E-01 0.0

-1.000000E-01 0.0

COMPENSATED TRANSFER FUNCTION GAIN = 1.000000E-00

COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E-00 1.000000E-00

COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
REAL PART IMAGINARY PART

-1.000000E-00 0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E-00 1.000000E-00

Computer Numerical Output Example 2

Figure IV-4E

COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS  
REAL PART  
IMAGINARY PART

1.00000000 0.0

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE  
TYPE. THEREFORE NO RIGHT-PLANE ZEROS WILL BE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

THE TOTAL NUMBER OF TRIALS CALLED TOP = 10000

THE COST FUNCTION TO BE USED IS THE TYPE 1

THE MINIMUM COST FUNCTION VALUE = 2.210733E-04

THE ERROR RETURN CODE FROM BOXPLX = 0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

9.830313E 01 2.297941E 02

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE:  
REAL PART  
IMAGINARY PART

-3.929228E-01 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

8.830558E 01 5.127121E 02

Computer Numerical Output Example 2

Figure IV-4F



OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE: IMAGINARY PART

-1.542545e-02 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 3.924033e-02

\*\*\*WARNING\*\*\*WARNING\*\*\*WARNING\*\*\*WARNING\*\*\*WARNING\*\*\*WARNING\*\*\*WARNING\*\*\*

THE PHASE SIMULATION OF THE SYSTEM WAS ABANDONED  
SINCE IT FAILS TO FIND SEPARATE ROOT FITTING SUBROUTINES  
FAILED TO FIND THE SYSTEM DENOMINATOR ROOTS

\*\*\*WARNING\*\*\*WARNING\*\*\*WARNING\*\*\*WARNING\*\*\*WARNING\*\*\*WARNING\*\*\*WARNING\*\*\*

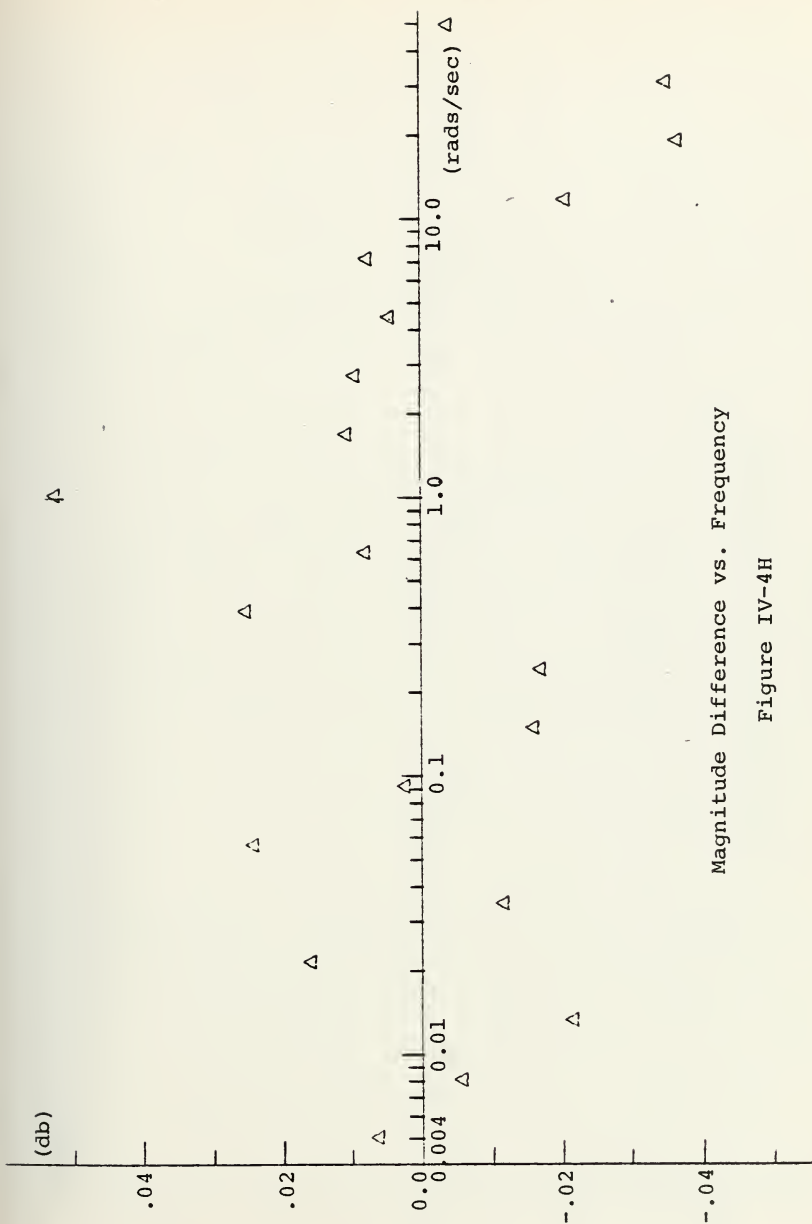
FREQUENCY	MAGNITUDE (DB)	DESIGN MAG (DB)	PHASE (DEG)	DESIGN PHASE
4.555555e-03	8.53133e 01	8.53133e 01	-1.072034	02
8.111111e-03	3.374274 01	3.374274 01	-1.166542	02
1.155555e-02	7.230137 01	7.230137 01	-1.267743	02
2.311111e-02	6.813500 01	6.813500 01	-1.413359	02
3.466666e-02	6.145034 01	6.145034 01	-1.514261	02
5.400000e-02	5.346347 01	5.346347 01	-1.571777	02
5.640000e-02	5.328258 01	5.328258 01	-1.583655	02
9.155555e-02	4.323354 01	4.323354 01	-1.550280	02
1.400000e-01	3.762142 01	3.762142 01	-1.374957	02
2.400000e-01	2.355332 01	2.355332 01	-1.373030	02
3.600000e-01	1.415451 01	1.415451 01	-1.256940	02
6.370000e-01	1.211531 00	1.260287 00	-1.217812	02
1.030000e-00	7.327664 00	3.665470 00	-1.271102	02
1.675555e-00	2.215930 00	3.665470 00	-1.284057	02
2.700000e-00	2.215930 00	-2.593234 00	-1.226572	02
4.400000e-00	2.215930 00	-3.373500 00	-1.644373	02
7.190000e-00	8.367465 00	-1.655421 01	-1.301681	02
1.170000e-01	5.57510 01	-2.655739 01	-2.156210	02
1.900000e-01	2.372550 01	-3.827280 01	-2.247270	02
3.069999e-01	3.330315 01	-5.035880 01	-2.476126	02
5.000000e-01	5.000000 01			02

THE ROOT TEST OF THE CHARACTERISTIC EQUATION INDICATES

THAT THE SYSTEM IS STABLE

Computer Numerical Output Example 2

Figure IV-4G



Magnitude Difference vs. Frequency

Figure IV-4H

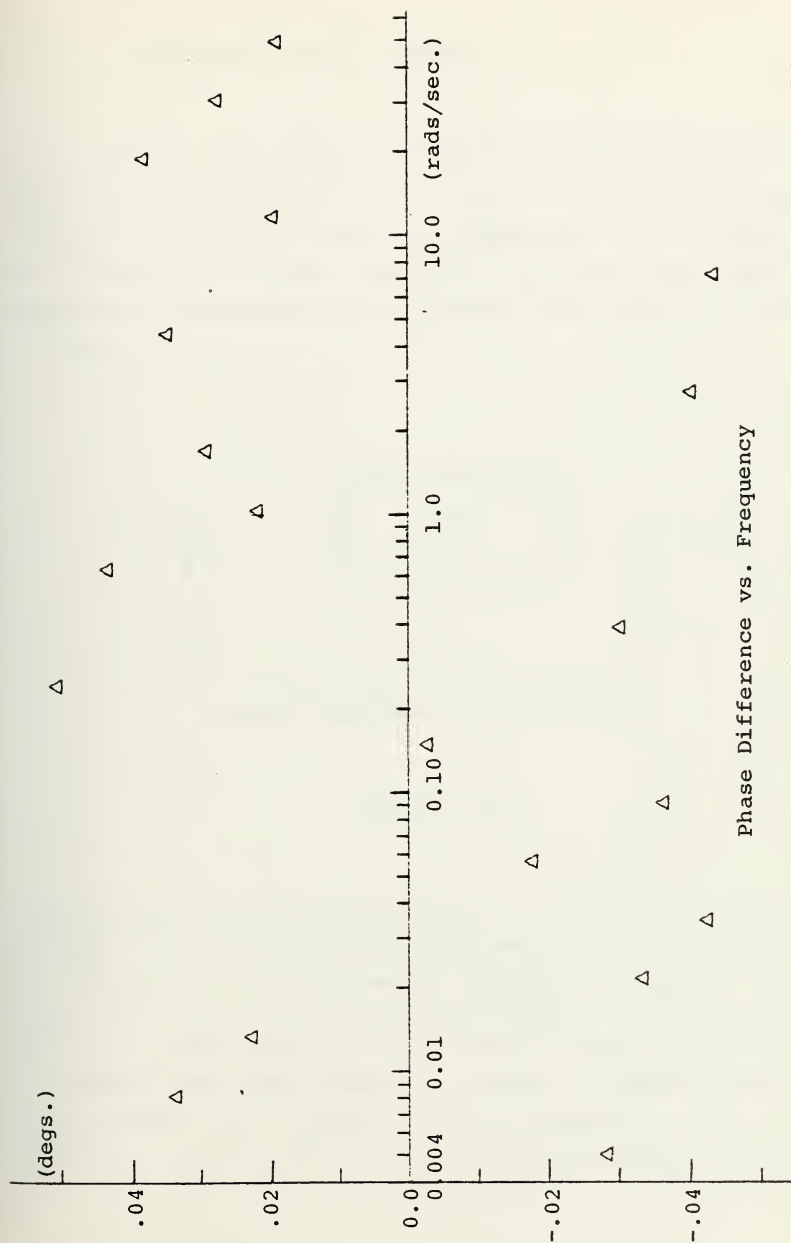
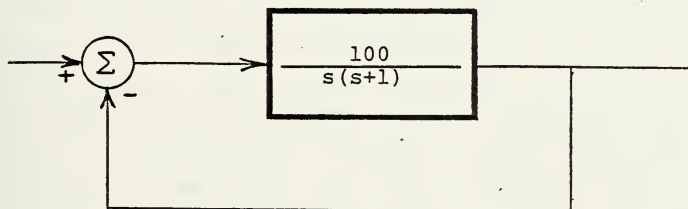


Figure IV-4I

### 3. Compensator Design, Example 3.

Another lag type compensation problem is presented in this example. This time, however, the existence of a fourth order polynomial for either the open loop system transfer function numerator or denominator has purposely been avoided to insure that the program will run to completion. The uncompensated system block diagram is shown in figure IV-5.



Design Example 3, Block Diagram

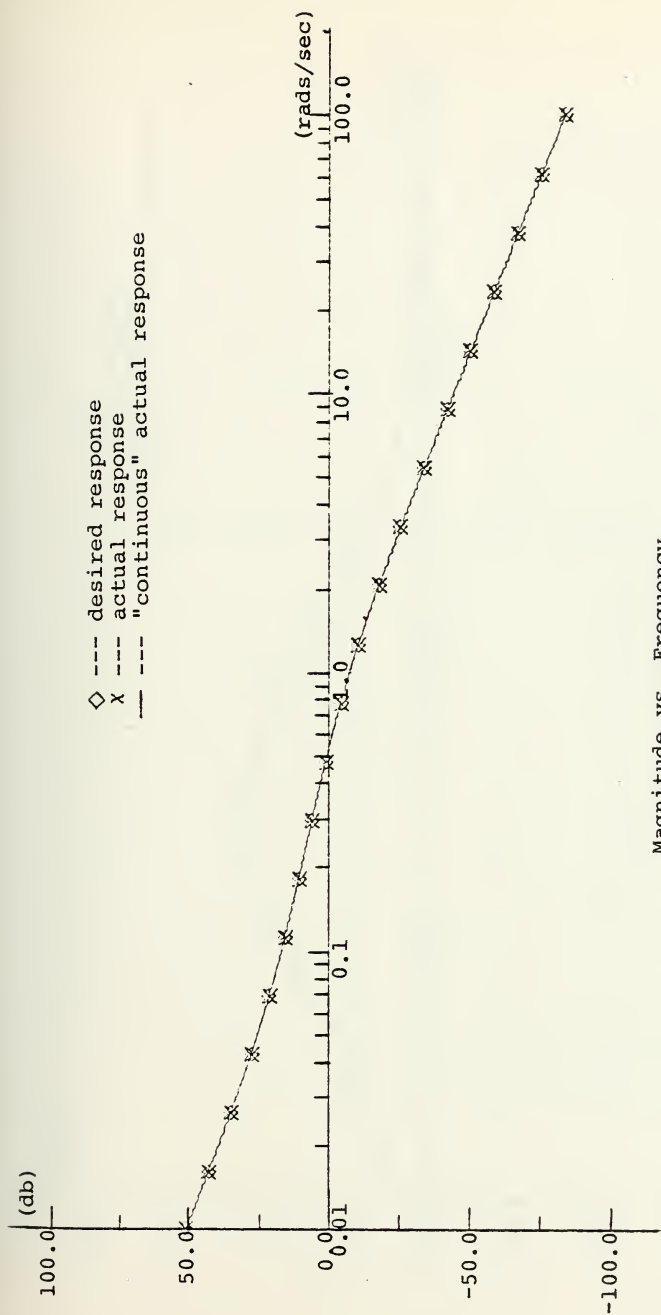
Figure IV-5

A Bode diagram for the uncompensated system along with a detailed discussion of the problem may be found in ref. 17. The uncompensated system does not possess an adequate phase margin to insure proper system performance. Thus, a single section lag compensator is to be used in order to achieve the desired open loop frequency response. Desired gain and phase profiles are selected for the compensated open loop frequency response using 20 discrete frequency points over the range from 0.01 to 100.0 radians. The profiles

selected along with the results achieved are shown in figures IV-5A, IV-5B, and IV-5C. The numerical values assumed as initial estimates of the compensator parameters and the numerical results of the output are shown in figures IV-5D, IV-5E, and IV-5F. Plots of the differences between the actual magnitude and phase, and the desired magnitude and phase are shown in figures IV-5G and IV-5H. In this particular problem approximately 1200 iterations of the minimization routine and 1 minute and 25 seconds were required for solution of the problem. In order to illustrate the idea that for minimum phase systems the compensator parameters may be determined to within only a fixed loss or gain when using only the phase profile, this same problem was solved again using the type 3 cost function which considers only the difference between the desired and actual phase at the discrete frequencies. The results are shown in figures IV-5I, IV-5J, and IV-5K. As can be seen from the magnitude plot in figure IV-5I, there is a constant gain error in the resulting magnitude curve. The numerical values assumed at the start and those returned after the minimization of the cost function are shown in the computer output of figures IV-5L, IV-5M, and IV-5N. The resultant magnitude response is higher than desired. The necessary correction that must be applied to obtain the desired response may be read directly from figure IV-5O, which shows the difference between the actual and desired magnitude response. That is, as can be seen from figure IV-5C, the resultant magnitude of the open loop system after compensation is approximately 36.8 db higher than what is desired. Since the program does not alter any of the plant parameters this means that the compensator gain value returned from the program when the type 3 cost function was used must be decreased by approximately a factor of 69.2. A comparison of the compensator gain values shown in figures IV-5F and IV-5N will show that they do indeed differ by this value. Also the astute observer may notice that there

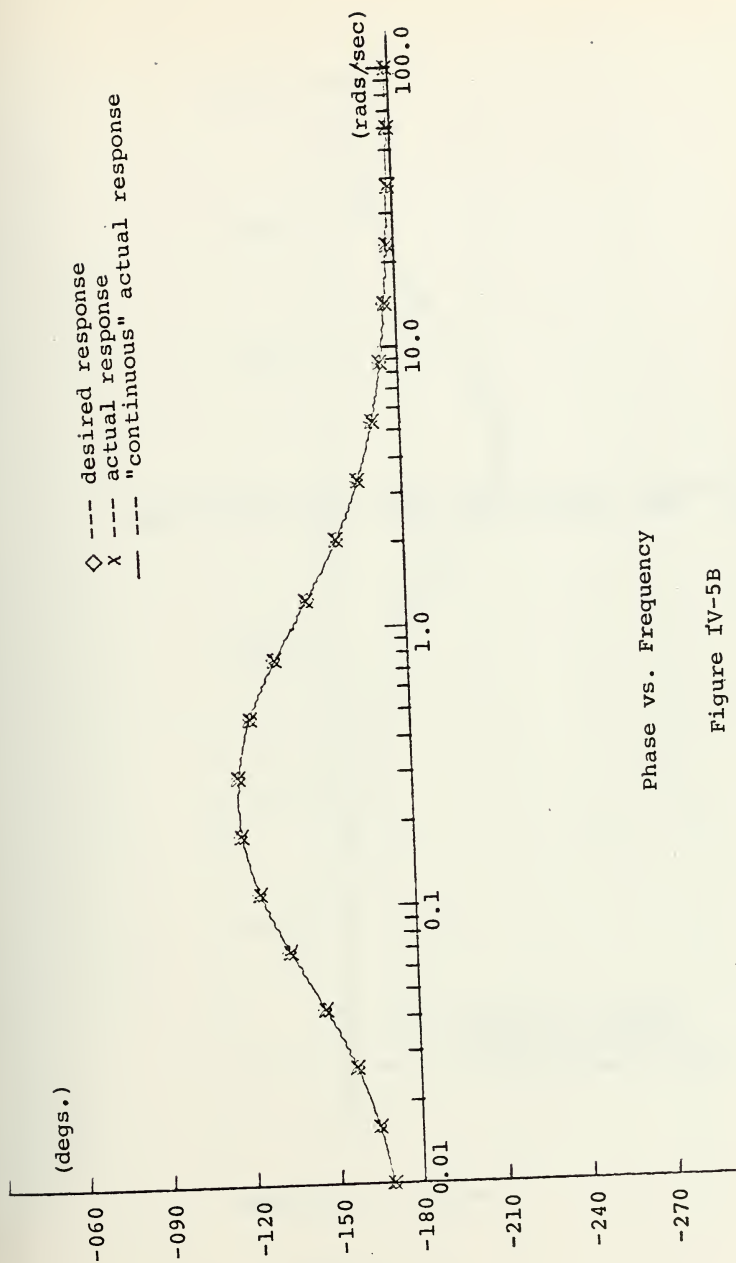
is a slight difference in the compensator pole value returned as the solution when the problem was solved using the type 3 cost function. While the difference in values is small it is suspected that in this particular case it is due to the fact that only a finite frequency range has been considered and the lower frequencies have been "slighted" somewhat in that the nonlinear phase curve has not yet flattened out at 0.1 radians. This is also the same situation that existed the first time the program was executed, but in this case the magnitude profile was also taken into account in the cost function and any deviations of the resultant magnitude values dominated the cost function value. In other words, it appears that in this particular problem the cost function is more sensitive to variations in the magnitude values than in the variations in the phase values.





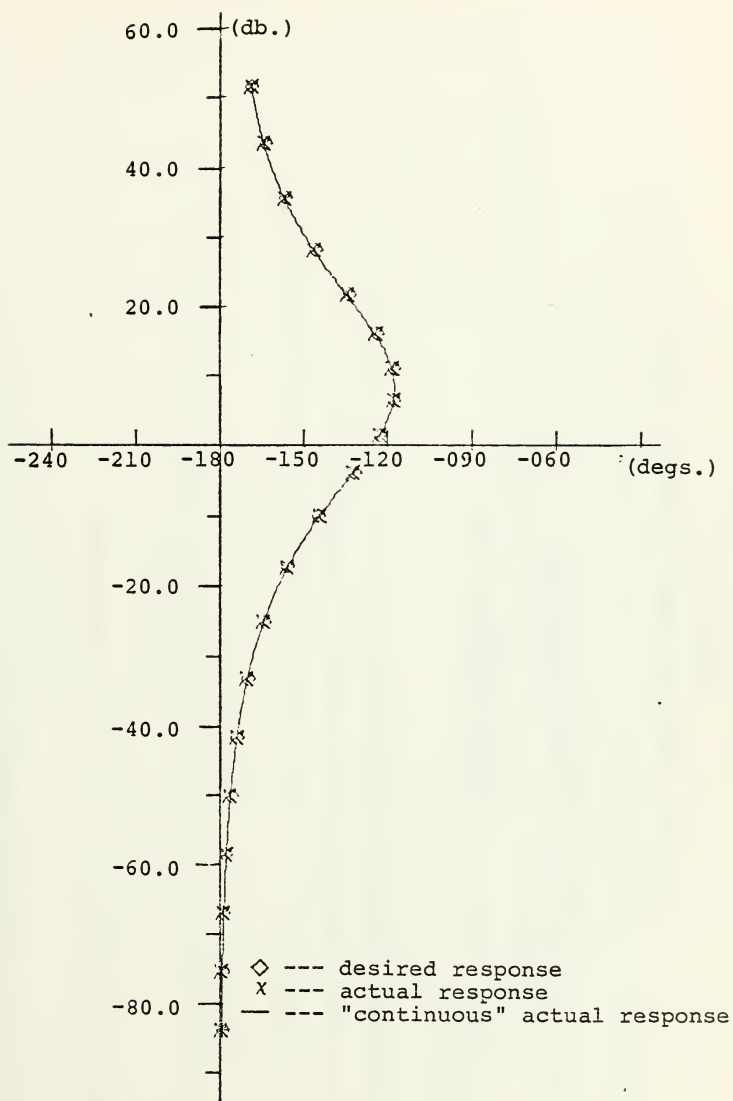
Magnitude vs. Frequency

Figure IV-5A



Phase vs. Frequency

Figure IV-5B



Magnitude vs. Phase

Figure IV-5C

TITLE --- COMPEN. OPTIMIZ. EXAMPLE 3 20PTS

UNCOMPENSATED TRANSFER FUNCTION GAIN = 1.000000E 02

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

0.0 1.000000E 00 1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
ARE: REAL PART IMAGINARY PART

0.0 0.0

-1.000000E 00 0.0

COMPENSATOR TRANSFER FUNCTION GAIN = 1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS  
ARE: REAL PART IMAGINARY PART

-1.000000E 00 0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 1.000000E 00

Computer Numerical Output Example 3, Run #1

Figure IV-5D

COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS  
ARE: REAL PART IMAGINARY PART

-----  
-1.000000E 00 0.0  
-----

-----  
THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE  
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION  
-----

-----  
THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000  
-----

-----  
THE COST FUNCTION IC BE USED IS THE TYPE 1  
-----

-----  
THE MINIMUM COST FUNCTION VALUE = 2.522382E-04  
-----

THE ERROR RETURN CODE FROM EXPLX = 0  
-----

-----  
OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S  
-----

1.622424E 00 3.05851E 01

-----  
OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
ROOTS ARE: REAL PART IMAGINARY PART

-----  
-5.590308E-02 0.0  
-----

-----  
OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S  
-----

2.066647E 00 4.652434E 03

Computer Numerical Output Example 3, Run #1  
Figure IV-5E

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE: REAL PART IMAGINARY PART

-4.26504E-04 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 0.252471E-03

FREQUENCY	MAGNITUDE (DB)	DESIRE MAG (DB)	PHASE (DEG)	DESIRE PHASE
5.595958E-03	5.18107E 01	5.159236E 01	-1.686534E 02	-1.690000E 02
1.640000E-02	4.133192E 01	4.134033E 01	-1.642871E 02	-1.640000E 02
2.280000E-02	3.525725E 01	3.540334E 01	-1.568000E 02	-1.570000E 02
4.345959E-02	2.147256E 01	2.799334E 01	-1.568000E 02	-1.460000E 02
1.130000E-01	1.472566E 01	2.151059E 01	-1.343227E 02	-1.340000E 02
1.830000E-01	1.267987E 01	1.588970E 01	-1.241552E 02	-1.240000E 02
2.580000E-01	1.057100E 01	1.095505E 01	-1.173781E 02	-1.180000E 02
4.650000E-01	6.357734E 00	6.235015E 00	-1.173781E 02	-1.180000E 02
7.650000E-01	-4.036615E 00	-4.026987E 00	-1.327559E 02	-1.320000E 02
1.270000E 00	-1.031600E 01	-1.037114E 01	-1.324644E 02	-1.320000E 02
2.370000E 00	-1.826250E 01	-1.758832E 01	-1.446464E 02	-1.450000E 02
3.600000E 00	-2.345973E 01	-2.549835E 01	-1.558401E 02	-1.560000E 02
5.455555E 00	-3.370943E 01	-3.383959E 01	-1.644401E 02	-1.640000E 02
8.660000E 00	-4.203107E 01	-4.203445E 01	-1.735451E 02	-1.730000E 02
1.440000E 01	-5.043342E 01	-5.042097E 01	-1.735451E 02	-1.730000E 02
2.335959E 01	-5.885547E 01	-5.861899E 01	-1.762640E 02	-1.760000E 02
3.785959E 01	-6.722279E 01	-6.725021E 01	-1.765984E 02	-1.760000E 02
6.155959E 01	-7.566531E 01	-7.565030E 01	-1.785784E 02	-1.780000E 02
1.000000E 02	-8.407538E 01	-8.408238E 01	-1.791251E 02	-1.790000E 02

THE ROOTS TEST OF THE CHARACTERISTIC EQUATION INDICATES  
THAT THE SYSTEM IS STABLE

Computer Numerical Output Example 3, Run #1

Figure IV-5F

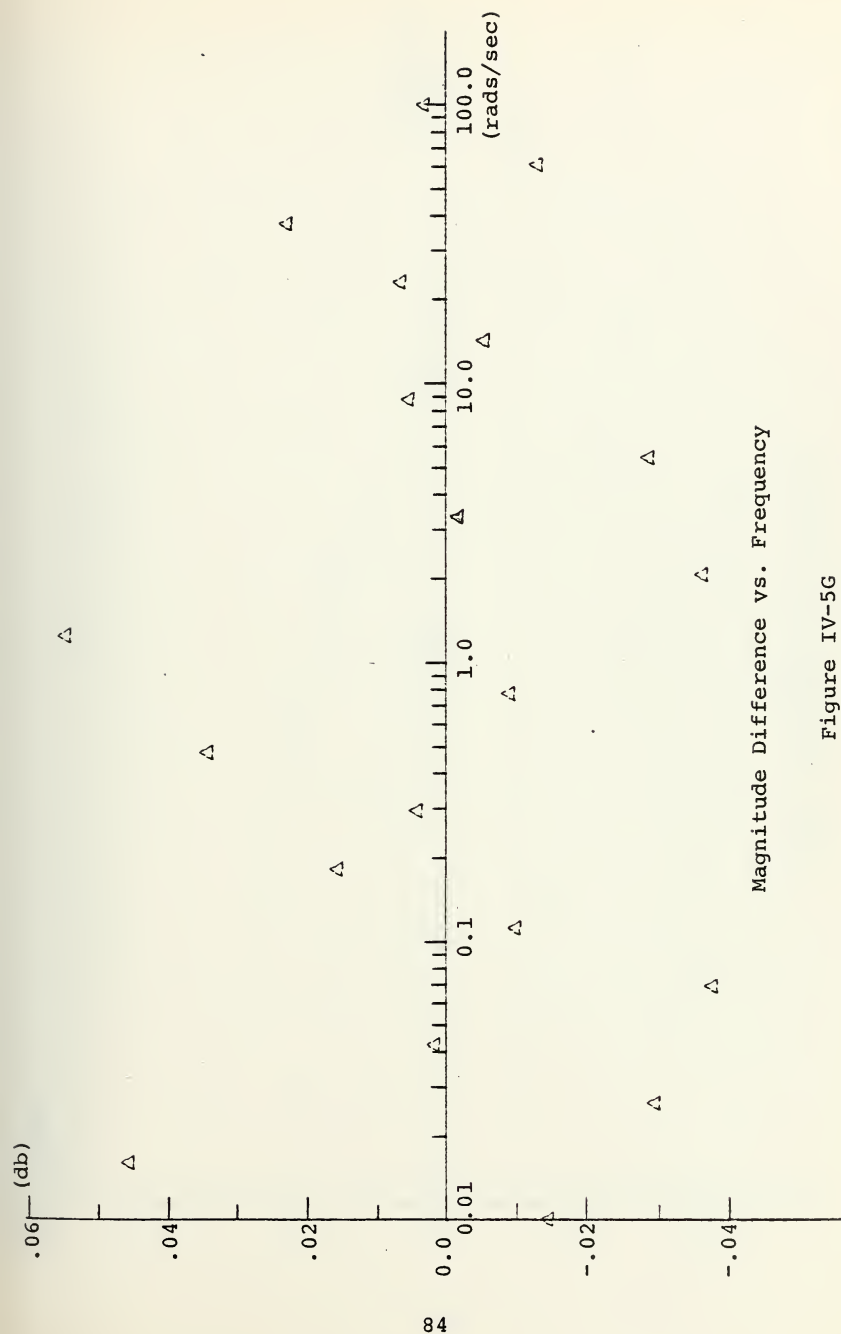
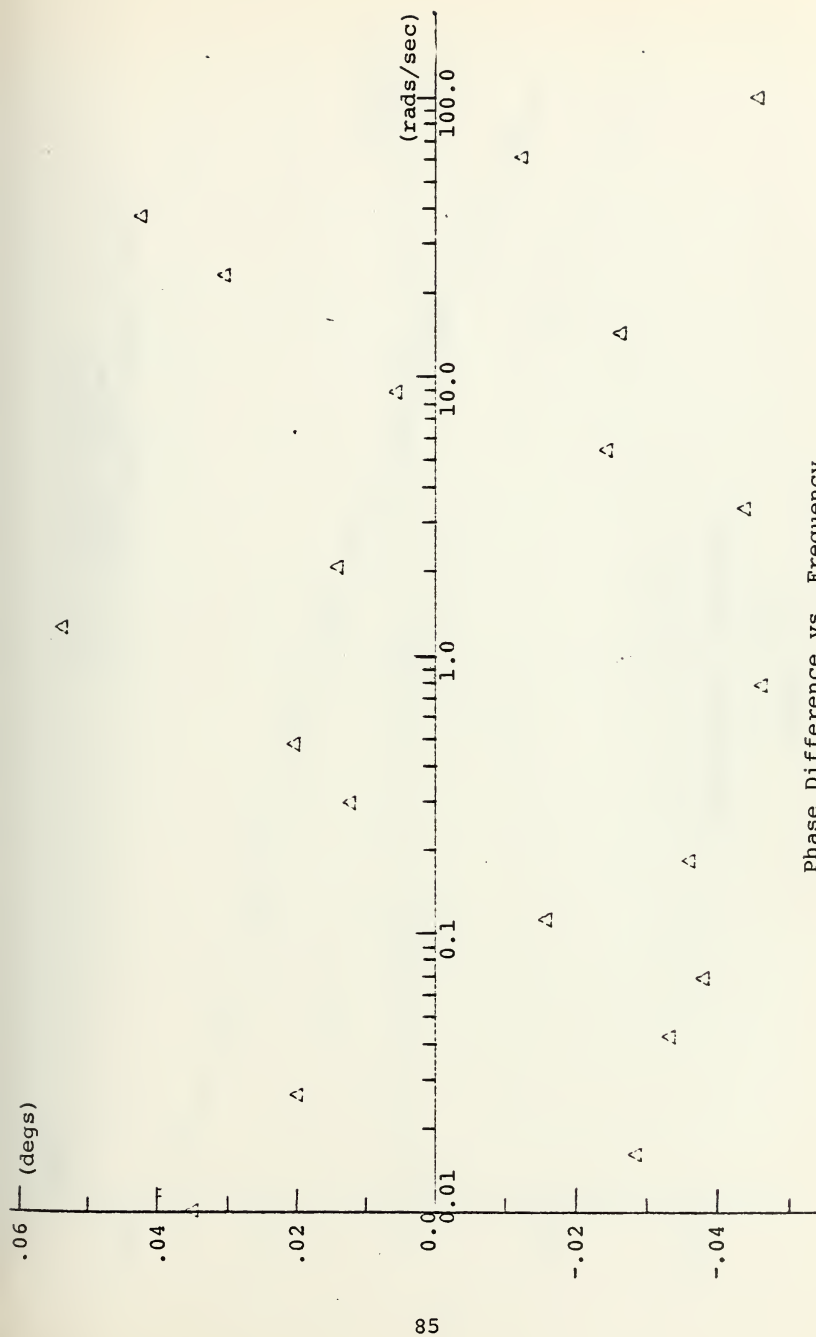


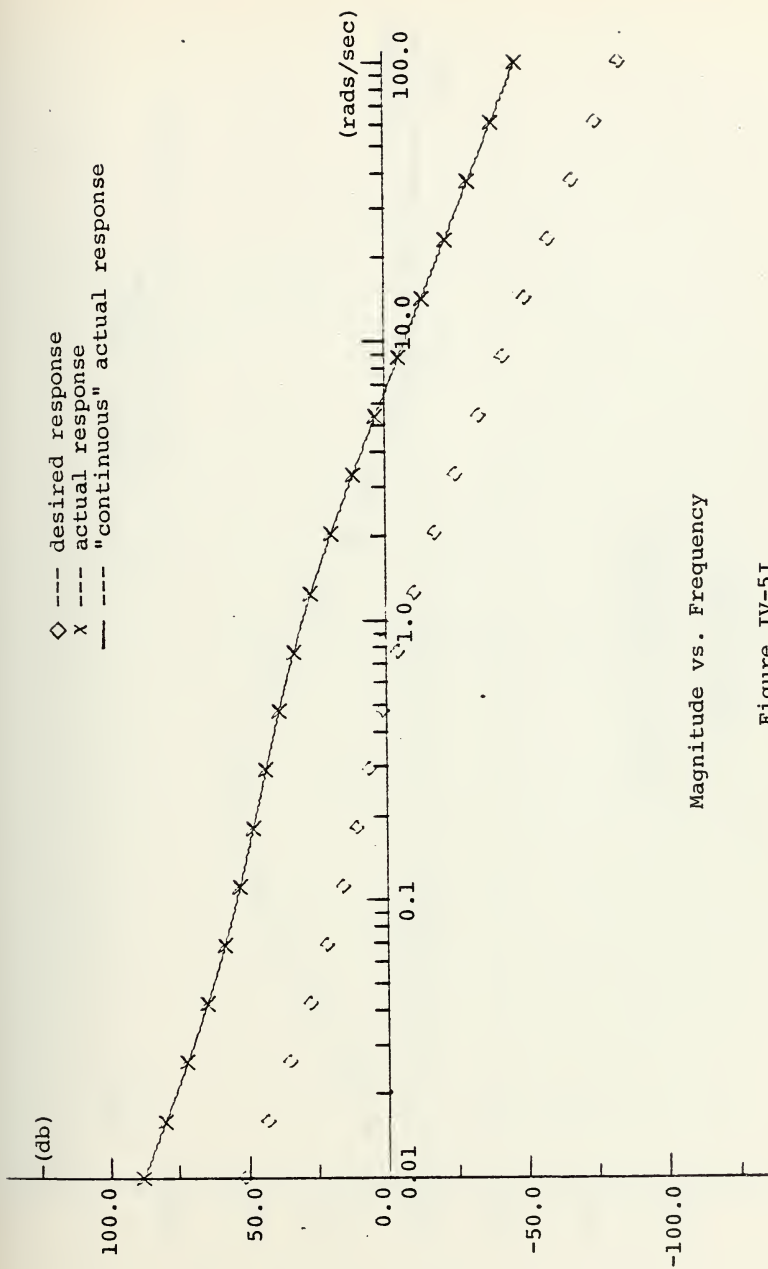
Figure IV-5G





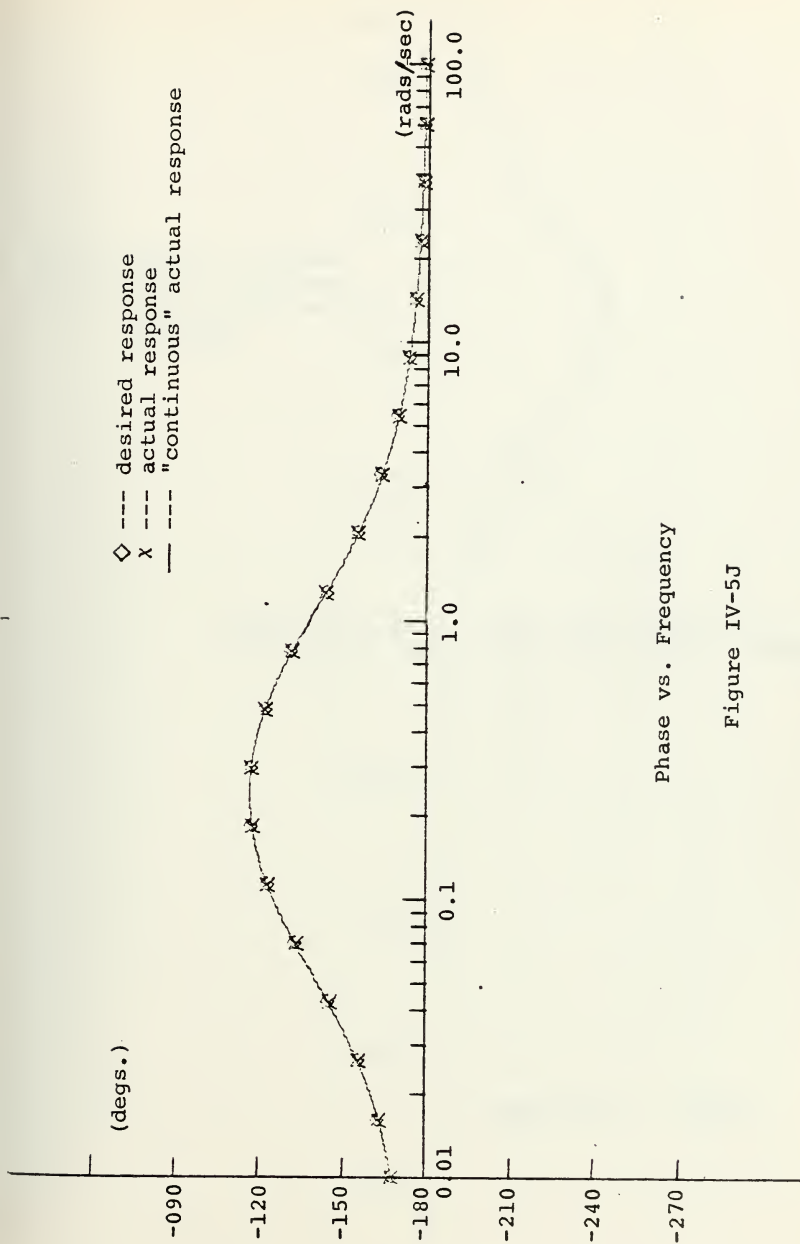
Phase Difference vs. Frequency

Figure IV-5H



Magnitude vs. Frequency

Figure IV-5I



Phase vs. Frequency

Figure IV-5J

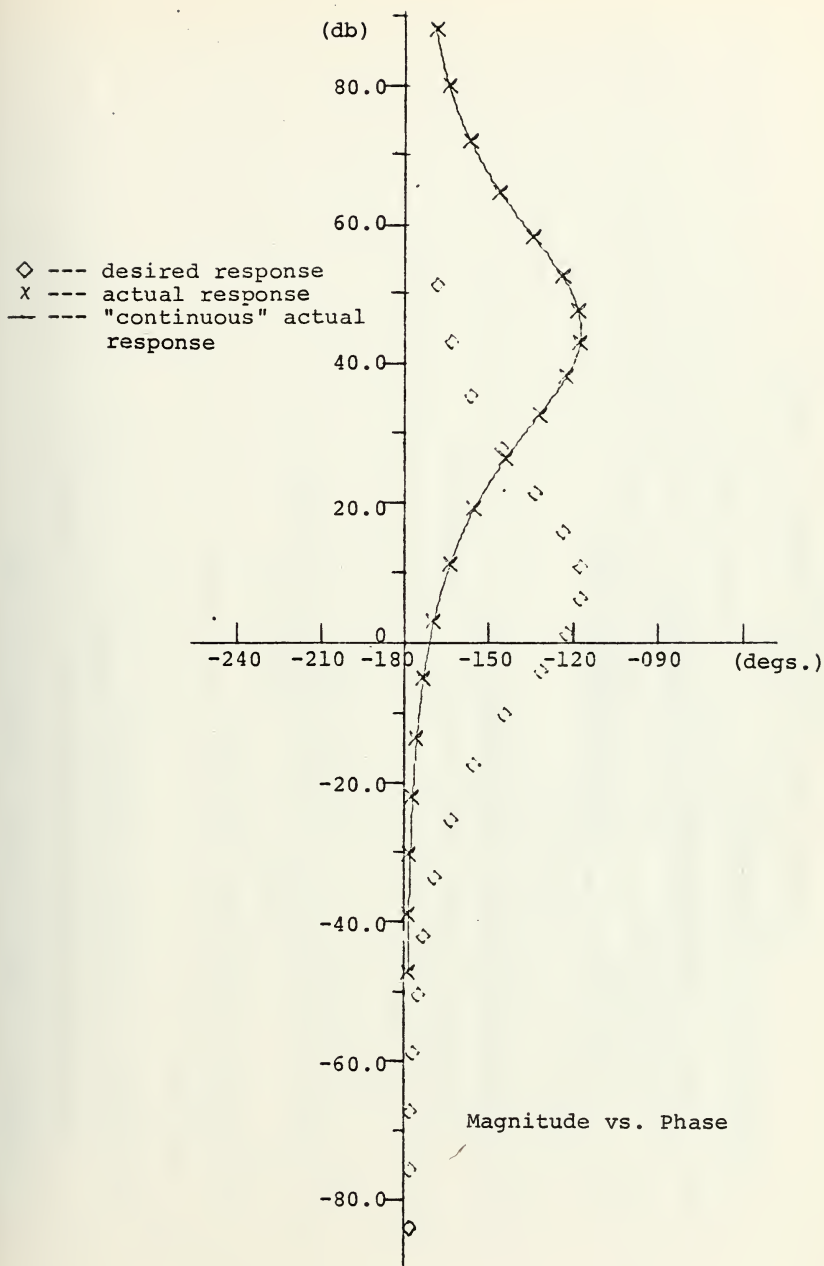


Figure IV-5K

TITLE --- CCMPEN. OPTIMIZ. EXAMPLE 3B 20PTS

UNCOMPENSATED TRANSFER FUNCTION GAIN = 1.000000E\_02

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

0.0 1.000000E 00 1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
ARE: REAL PART IMAGINARY PART

0.0 0.0

-1.000000E 00 0.0

COMPENSATOR TRANSFER FUNCTION GAIN = 1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR  
ARE: REAL PART IMAGINARY PART

-1.000000E 00 0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 1.000000E 00

Computer Numerical Output Example 3, Run #2

Figure IV-5L

COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS

ARE:

REAL PART

IMAGINARY PART

-1.000000E 00 0.0

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE  
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

THE COST FUNCTION TO BE USED IS THE TYPE 3

THE MINIMUM COST FUNCTION VALUE = 7.442613E-05

THE ERROR RETURN CODE FROM EXPLX = 0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

7.643527E 03 1.341418E 05

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE:

REAL PART

IMAGINARY PART

-5.921741E-02 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.122473E 02 3.089048E 05

Computer Numerical Output Example 3, Run #2  
Figure IV-5M

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE: REAL PART IMAGINARY PART

-3.666091E-C4 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 4.342495E-01

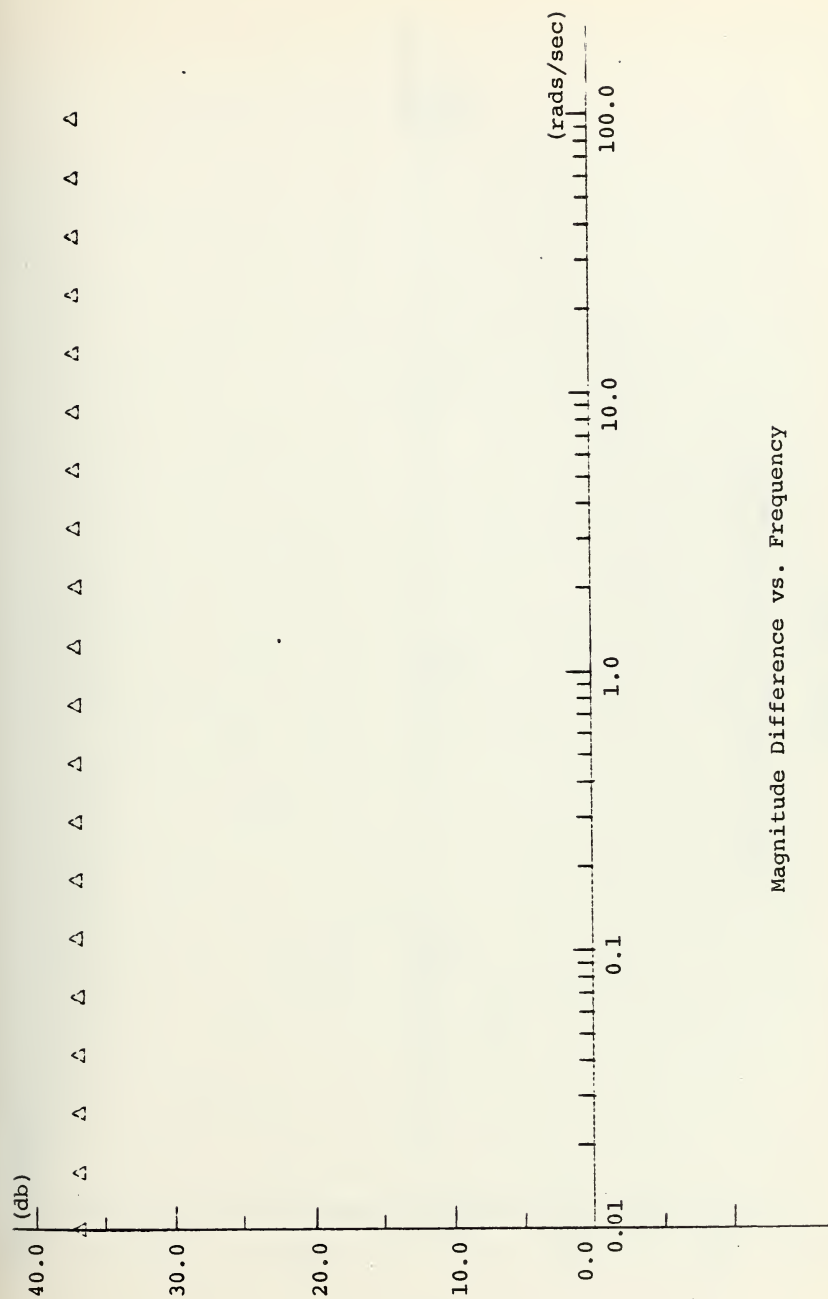
FREQUENCY	MAGNITUDE (DB)	DESIRED MAG (DB)	PHASE (DEG)	DESIRED PHASE
5.999998E-03	8.011933E 01	5.159566E 01	-1.688882E 02	-1.690000E 02
1.620000E-02	7.011323E 01	4.334634E 01	-1.682323E 02	-1.640000E 02
2.640000E-02	7.212205E 01	3.540230E 01	-1.565867E 02	-1.570000E 02
4.280000E-02	6.476300E 01	2.799347E 01	-1.461021E 02	-1.460000E 02
6.545557E-02	5.826918E 01	2.151093E 01	-1.341061E 02	-1.340000E 02
1.130000E-01	5.269189E 01	1.588976E 01	-1.231180E 02	-1.240000E 02
1.836000E-01	4.775215E 01	1.095542E 01	-1.123169E 02	-1.180000E 02
2.830000E-01	4.306215E 01	6.235075E 00	-1.177628E 02	-1.180000E 02
4.830000E-01	3.823300E 01	1.363709E 00	-1.221269E 02	-1.230000E 02
7.830000E-01	3.275771E 01	-4.026587E 00	-1.324192E 02	-1.320000E 02
1.270000E 00	2.651689E 01	-1.037111E 01	-1.444362E 02	-1.450000E 02
2.270000E 00	1.520837E 01	-1.758252E 01	-1.584366E 02	-1.560000E 02
3.360000E 00	1.133332E 01	-2.549309E 01	-1.732387E 02	-1.640000E 02
5.455555E 00	3.124622E 00	-3.363059E 01	-1.732387E 02	-1.700000E 02
8.660000E 00	-5.136035E 01	-4.203645E 01	-1.732387E 02	-1.700000E 02
1.440000E 01	-1.360353E 01	-5.042867E 01	-1.732387E 02	-1.760000E 02
2.335559E 01	-2.202173E 01	-5.886189E 01	-1.776968E 02	-1.760000E 02
3.785559E 01	-3.039317E 01	-6.725021E 01	-1.776968E 02	-1.780000E 02
6.155559E 01	-3.982957E 01	-7.565030E 01	-1.791245E 02	-1.790000E 02
1.000000E 02	-4.724564E 01	-8.403233E 01	-1.794607E 02	-1.790000E 02

THE ROOTS TEST OF THE CHARACTERISTIC EQUATION INDICATES  
THAT THE SYSTEM IS STABLE

Computer Numerical Output Example 3, Run #2

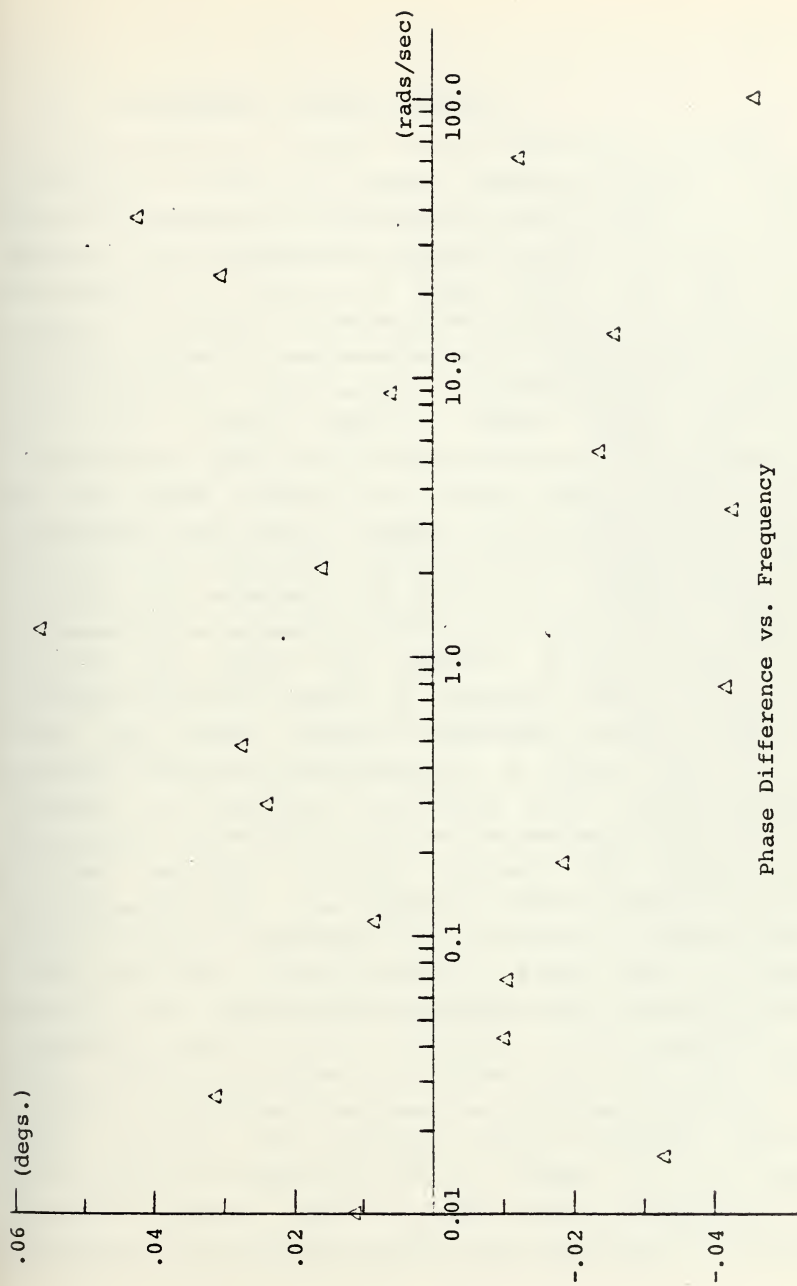
Figure IV-5N





Magnitude Difference vs. Frequency

Figure IV-50



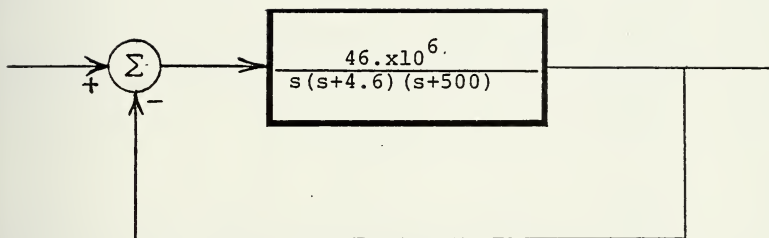
Phase Difference vs. Frequency

Figure IV-5P

#### 4. Compensator Design, Example 4

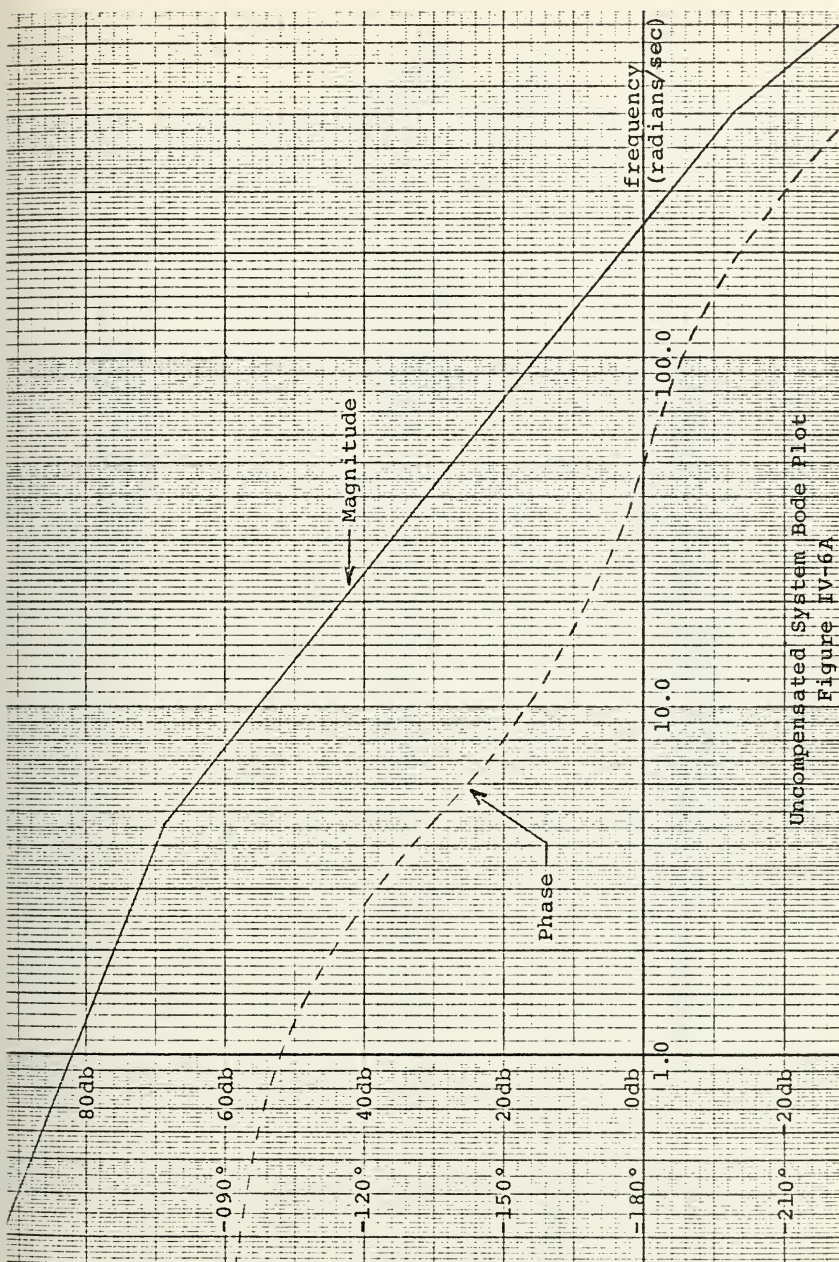
In this example a slightly more complicated problem is presented, in that the final solution requires a double section compensator of the form of a notch filter. A block diagram of the uncompensated system is shown in figure IV-6. As discussed by Thaler and Brown [18], from whom this example was taken, the uncompensated system is unstable and the Bode diagram of the uncompensated system shown in figure IV-6A indicates a phase margin of approximately -30 degrees. With such a large negative phase margin for the uncompensated system and the rapid drop in slope of both the gain and phase curves the designer might suspect that compensation using only a single section compensator will be difficult to accomplish. Again desired magnitude and phase profiles are selected and initially a single section of compensation is assumed to observe how close to the desired response the system will perform. As can be seen in the resultant graphical output of figures IV-6A, IV-6B, and IV-6C, the single section compensator fails to meet the required frequency specifications. In fact, while it is possible to stabilize this system with a single section compensator the bandwidth will be excessively large and the desired magnitude profile prevents the program from accomplishing this. The numerical results returned from the single section compensator run are shown in figures IV-6D, IV-6E, and IV-6F. As can be seen in figure IV-6F, the Routh test of the characteristic equation also indicates system instability which is to be expected. The magnitude and phase difference curves of figures IV-6G and IV-6H, which indicate the difference between the specified magnitude and phase profiles and the values actually achieved, indicate that additional compensation may be needed at the higher frequency ranges. Thus a double section compensator was assumed and the results achieved are illustrated in figures

IV-6I, IV-6J, and IV-6K. The program is able to satisfy the specifications with a double section of compensation and the parameters required to accomplish this are shown in figures IV-6I, IV-6L, and IV-6N. The differences between the desired and specified magnitude and phase values for the double section of compensation inserted in the system are shown in figures IV-6O and IV-6P.

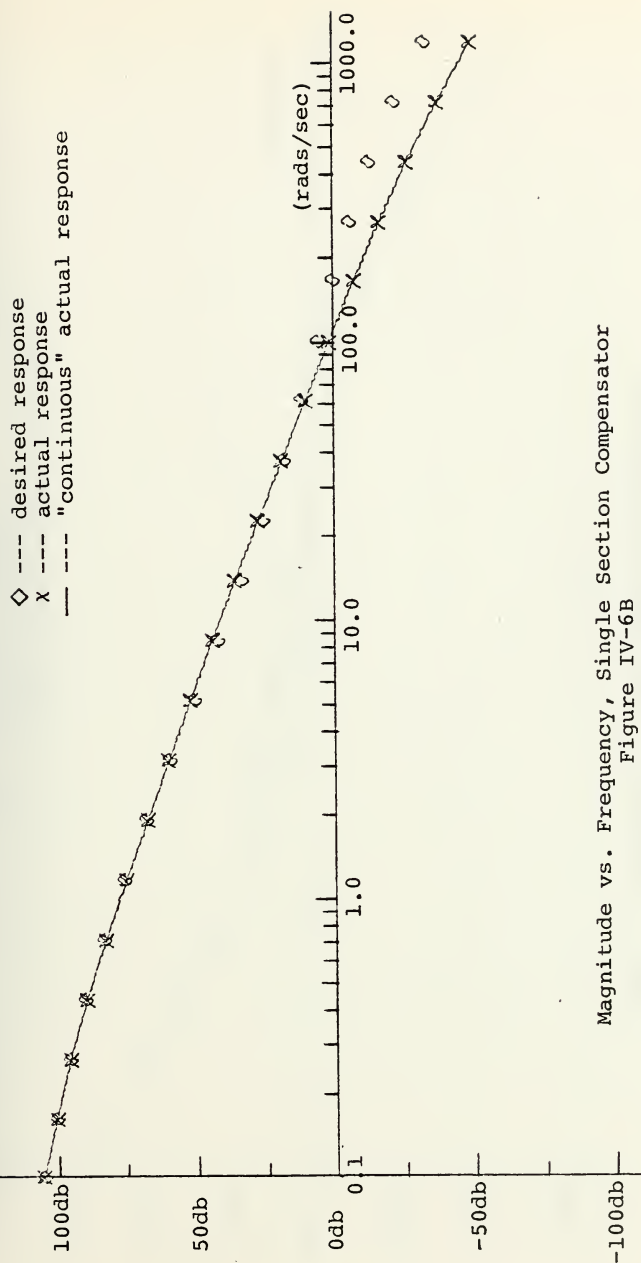


Design Example 4, Block Diagram

Figure IV-6

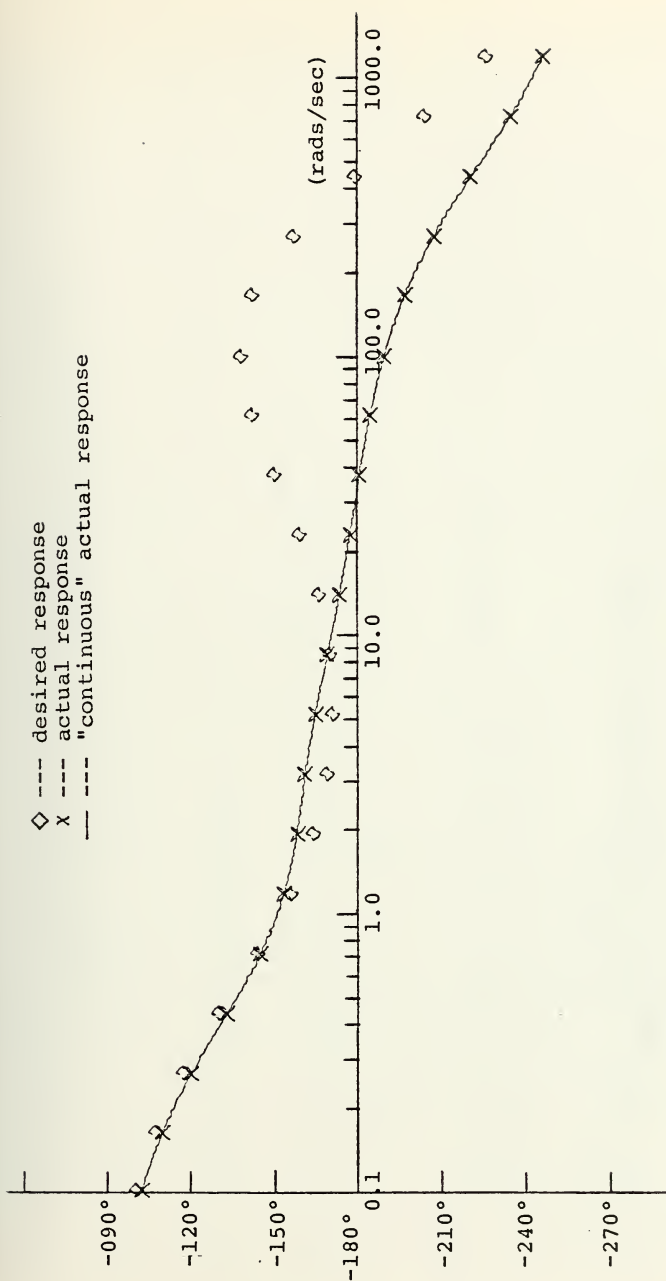


Uncompensated System Bode Plot  
Figure IV-6A



Magnitude vs. Frequency, Single Section Compensator  
Figure IV-6B





Phase vs. Frequency, Single Section Compensator  
Figure IV-6C

TITLE --- COMP. EXP. 4 POLES TYPE 1 CST SH

UNCOMPENSATED TRANSFER FUNCTION GAIN = 4.60000E 07

UNCOMPENSATED TRANSFER FUNCTION COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR COEFFICIENTS IN ASCENDING POWERS OF S

0.0 2.26000E 02 5.04500E 02 1.00000E 00

ARE: REAL PART IMAGINARY PART

0.0 0.0

-4.598595E 00 0.0

-5.00000E 02 0.0

COMPENSATOR TRANSFER FUNCTION GAIN = 1.00000E 00

COMPENSATOR TRANSFER FUNCTION COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 1.00000E 00

COMPENSATOR TRANSFER FUNCTION DENOMINATOR COEFFICIENTS IN ASCENDING POWERS OF S

-1.000000E 00 0.0

COMPENSATED TRANSFER FUNCTION DENOMINATOR COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 1.00000E 00

Computer Numerical Output  
Figure IV-6D

AG: COMPARATIVE TRIANGLE FUNCTION ON SEQUENTIAL ROOTS  
FEAL PART IMAGINARY PART

-1.00000000 00 0.0

THE COMPARATIVE TRIANGLE FUNCTION IS OF THE MINIMUM PHASE  
TYPE, THE FIFTY EIGHT HALF PLANE ZEROS WILL BE ALLOWED TO  
THE SOLUTION FOR THE COMPARATIVE TRIANGLE FUNCTION

THE TOTAL NUMBER OF TRIANGLES CALLED FOR = 10000

THE COST FUNCTION TO BE USED IS THE TYPE 1

THE FIRST VERTEX VALUES ARE:

5.593376E-05 1.026201E-05 3.29276E-05 1.00135E-05

THE MINIMUM COST FUNCTION VALUE = 5.021001E-00

THE ERROR RETURN CODE FROM THE XOLX = 2

OPTIMIZED COMPARATIVE TRIANGLE FUNCTION MINIMUM VALUE  
COEFFICIENTS IN ASCENDING ORDER OF S

5.593376E-05 1.026201E-05

Computer Numerical Output  
Figure IV-6E

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
ROOTS ARE: REAL PART IMAGINARY PART

-3.12361E-01 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

3.292767E-05 7.00155E-05

OPTIMIZED COMPENSATOR TRANSFER FUNCTION POLYMERIZATION  
ROOTS ARE: REAL PART IMAGINARY PART

-4.220654E-01 0.0

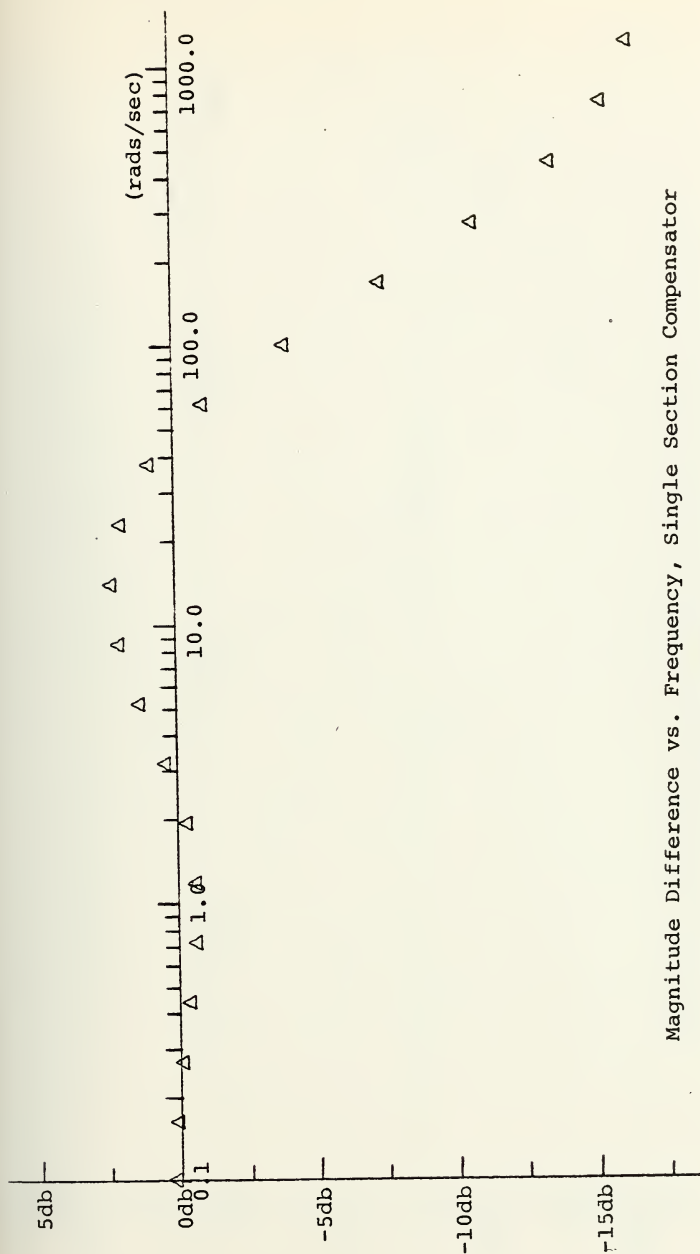
OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 1.00238E-01

\*\*\*\*\* W A K N I T C \*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*

THE CONTROL SYSTEM IS A CHEMICAL REACTION SYSTEM  
INSTABILITY

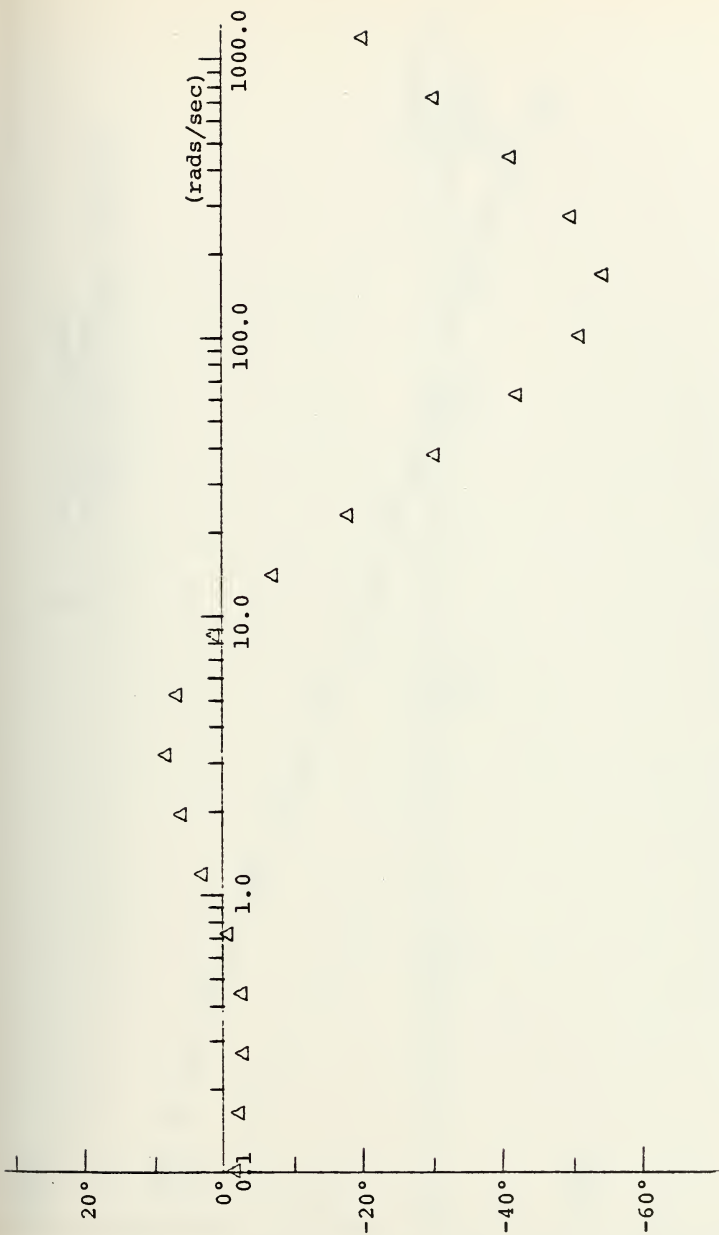
\*\*\*\*\* W A K N I T C \*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*

Computer Numerical Output  
Figure IV-6F

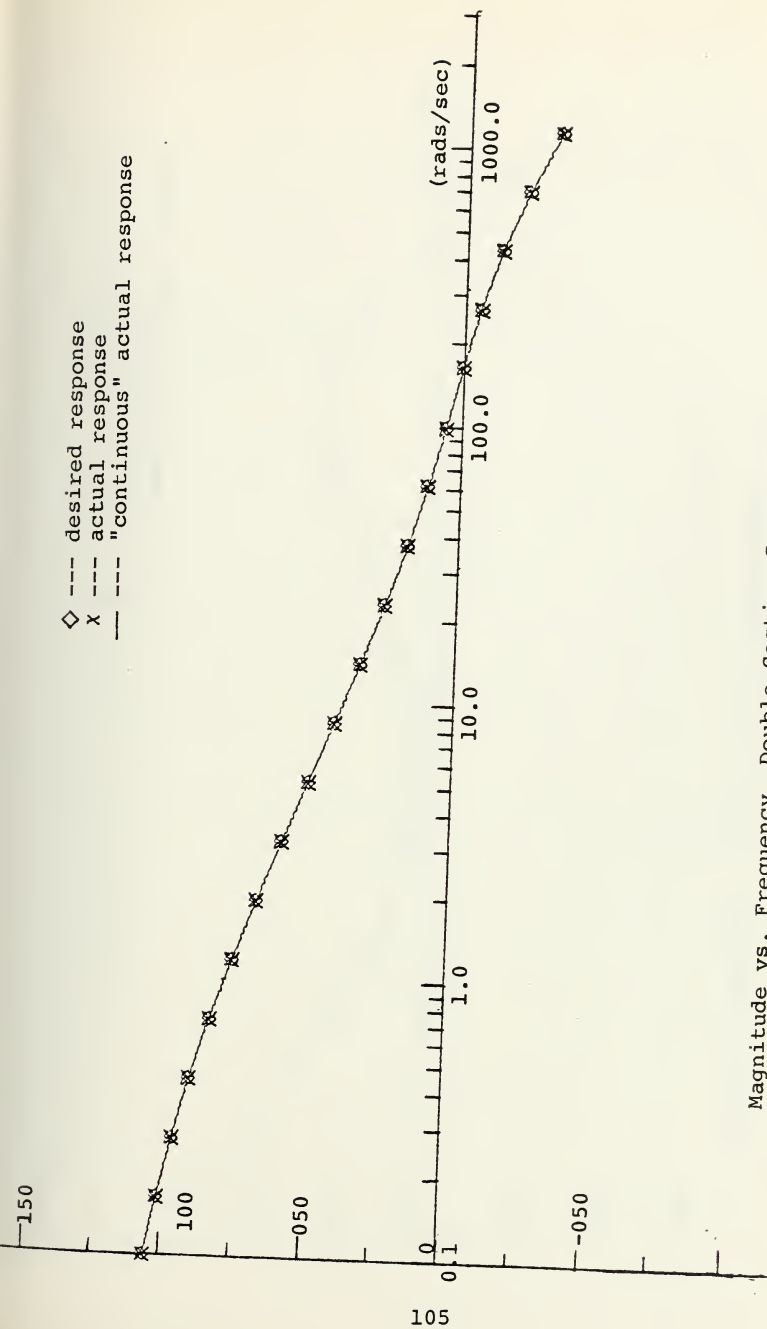


Magnitude Difference vs. Frequency, Single Section Compensator

Figure IV-6G

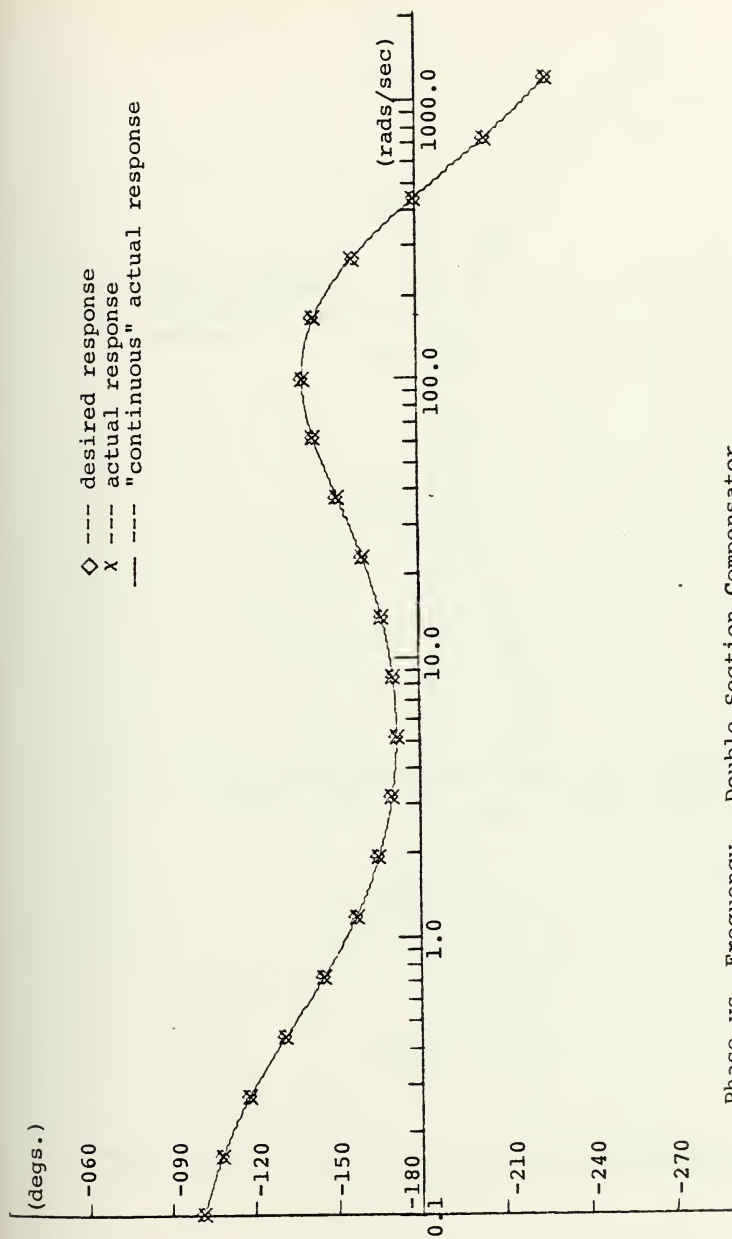


Phase Difference vs. Frequency, Single Section Compensator  
Figure IV-6H



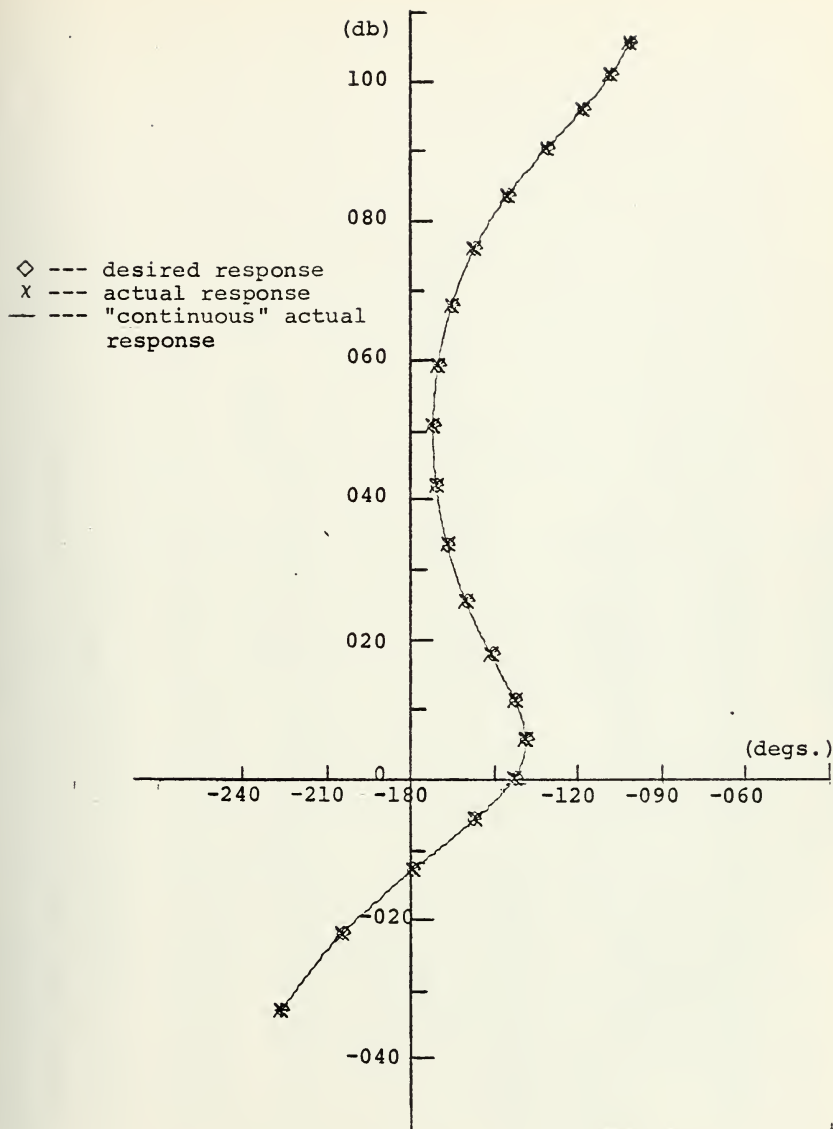
Magnitude vs. Frequency, Double Section Compensator  
Figure IV-61





Phase vs. Frequency, Double Section Compensator

Figure IV-6J



Magnitude vs. Phase, Double Section Compensator  
Figure IV-6K

TITLE --- CCMPD. UNIT. 182. 4. 20.015. TYPE 1 CSR.

UNCOMPENSATED TRANSFER FUNCTION GAIN = 4.00000E 07

UNCOMPENSATED TRANSFER FUNCTION COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR COEFFICIENTS IN ASCENDING POWERS OF S

0.0 2.40000E 03 5.04599E 02 1.00000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR PARTIAL FRACTIONAL PART

0.0 0.0

-4.59949E 00 0.0

-5.00000E 02 0.0

COMPENSATOR TRANSFER FUNCTION GAIN = 1.00000E 00

COMPENSATOR TRANSFER FUNCTION COEFFICIENTS IN ASCENDING POWERS OF S

1.00000E 00 2.00000E 00 1.00000E 00

COMPENSATOR TRANSFER FUNCTION DENOMINATOR PARTIAL FRACTIONAL PART

-1.00000E 00 0.0

-1.00000E 00 0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR COEFFICIENTS IN ASCENDING POWERS OF S

1.00000E 00 2.00000E 00 1.00000E 00

Computer Numerical Output  
Figure IV-6L

COMPENSATOR TRANSFER FUNCTION COEFFICIENT ROOTS  
REAL PART IMAGINARY PART

-1.00000E 00 0.0

-1.00000E 00 0.0

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE  
TYPE, THEREFORE NO RIGHT-HALF-PLANE ZEROS WILL BE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

THE COST FUNCTION TYPE USED IS TYPE 1

THE BEST VERTEX VALUES ARE:

4.906714E 05	1.042394E 05	1.054592E 03	4.907928E 05	9.853190E 05
1.563274E 03				

THE MINIMUM COST FUNCTION VALUE = 1.532374E-04

THE TOTAL PERFORMED COMPLEX POINTS = 0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION COEFFICIENTS  
COEFFICIENTS IN ASCENDING POWERS OF S

4.906714E 05	1.042394E 05	1.054592E 03
--------------	--------------	--------------

Computer Numerical Output  
Figure IV-6M

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
ROOTS ARE:

-5.0139241 01 0.0

-4.9812772 00 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

4.907023E 05 9.352190 05 1.963274 04

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE:

-5.0139241 02 0.0

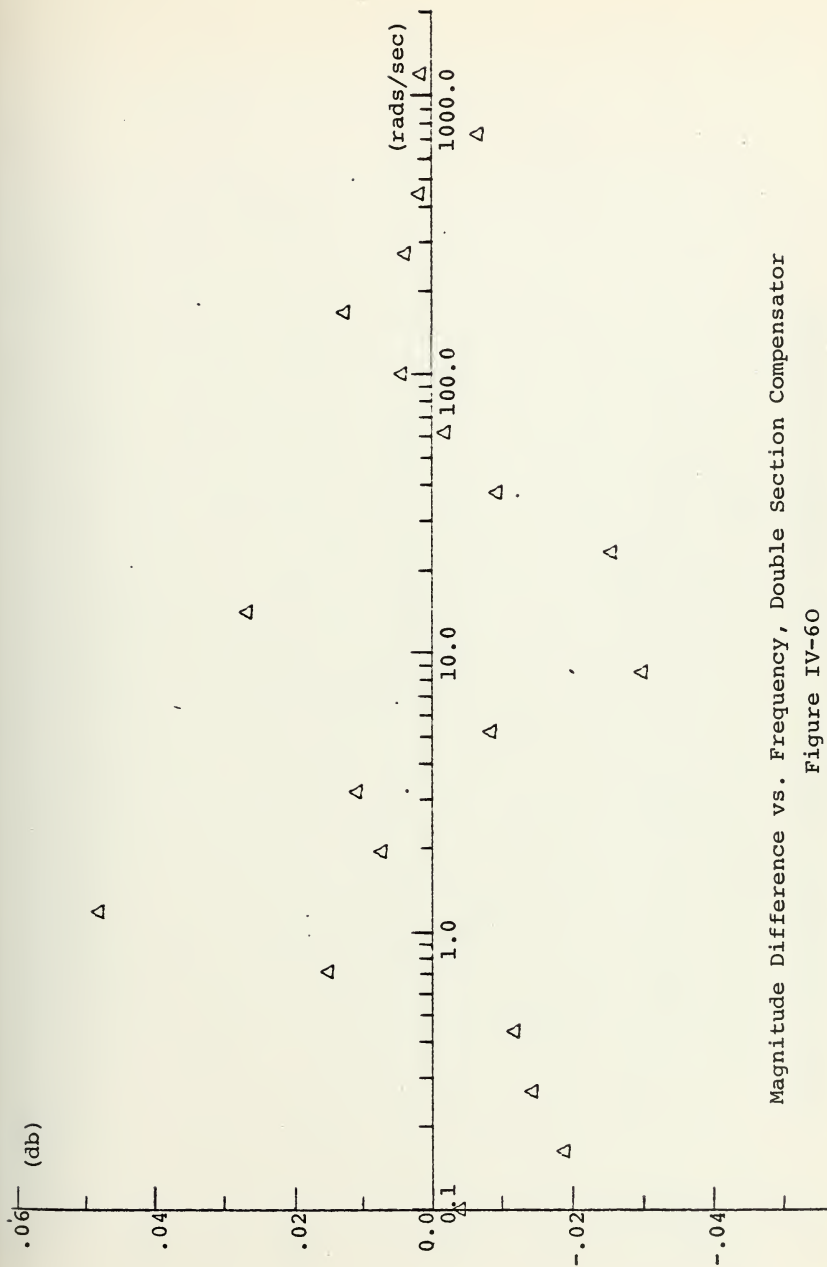
-4.985962E-01 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 1.000670E 00

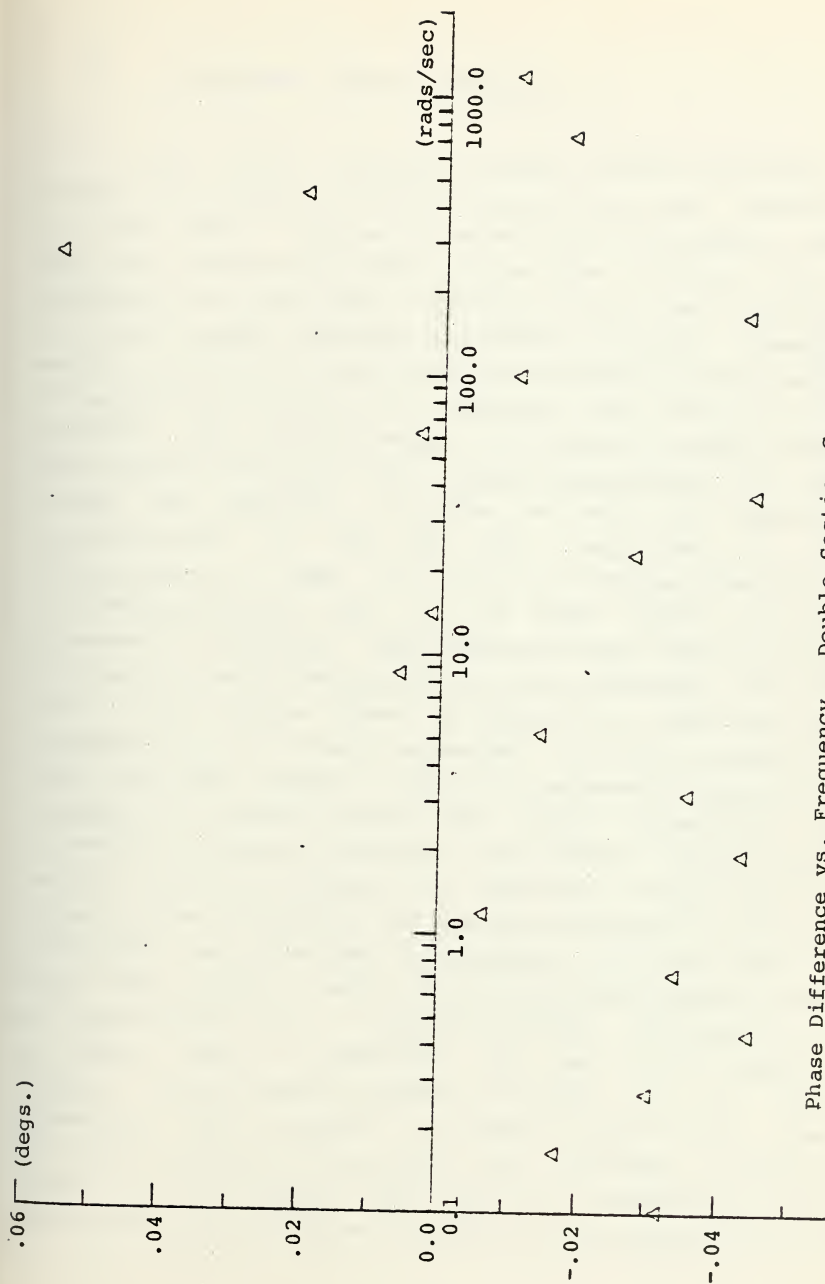
FREQUENCY	NUMERATOR (DB)	DENOMINATOR (DB)	PHASE (DEG)	DESIGN PHASE
0.999999E-02	1.000000E+00	1.000000E+00	-1.013440	02
1.000000E-01	1.000000E+00	1.000000E+00	-1.013440	02
1.000000E-01	1.000000E+00	1.000000E+00	-1.013440	02
2.000000E-01	1.000000E+00	1.000000E+00	-1.013440	02
4.000000E-01	1.000000E+00	1.000000E+00	-1.013440	02
6.000000E-01	1.000000E+00	1.000000E+00	-1.013440	02
8.000000E-01	1.000000E+00	1.000000E+00	-1.013440	02
1.000000E+00	1.000000E+00	1.000000E+00	-1.013440	02
1.200000E+00	1.000000E+00	1.000000E+00	-1.013440	02
1.400000E+00	1.000000E+00	1.000000E+00	-1.013440	02
1.600000E+00	1.000000E+00	1.000000E+00	-1.013440	02
1.800000E+00	1.000000E+00	1.000000E+00	-1.013440	02
2.000000E+00	1.000000E+00	1.000000E+00	-1.013440	02
2.200000E+00	1.000000E+00	1.000000E+00	-1.013440	02
2.400000E+00	1.000000E+00	1.000000E+00	-1.013440	02
2.600000E+00	1.000000E+00	1.000000E+00	-1.013440	02
2.800000E+00	1.000000E+00	1.000000E+00	-1.013440	02
3.000000E+00	1.000000E+00	1.000000E+00	-1.013440	02
3.200000E+00	1.000000E+00	1.000000E+00	-1.013440	02
3.400000E+00	1.000000E+00	1.000000E+00	-1.013440	02
3.600000E+00	1.000000E+00	1.000000E+00	-1.013440	02
3.800000E+00	1.000000E+00	1.000000E+00	-1.013440	02
4.000000E+00	1.000000E+00	1.000000E+00	-1.013440	02
4.200000E+00	1.000000E+00	1.000000E+00	-1.013440	02
4.400000E+00	1.000000E+00	1.000000E+00	-1.013440	02
4.600000E+00	1.000000E+00	1.000000E+00	-1.013440	02
4.800000E+00	1.000000E+00	1.000000E+00	-1.013440	02
5.000000E+00	1.000000E+00	1.000000E+00	-1.013440	02

THE FOURTH TEST OF THE CHARACTERISTIC EQUATION INDICATES  
THAT THE SYSTEM IS STABLE

Figure IV-6N



Magnitude Difference vs. Frequency, Double Section Compensator  
Figure IV-60



Phase Difference vs. Frequency, Double Section Compensator  
Figure IV-6p

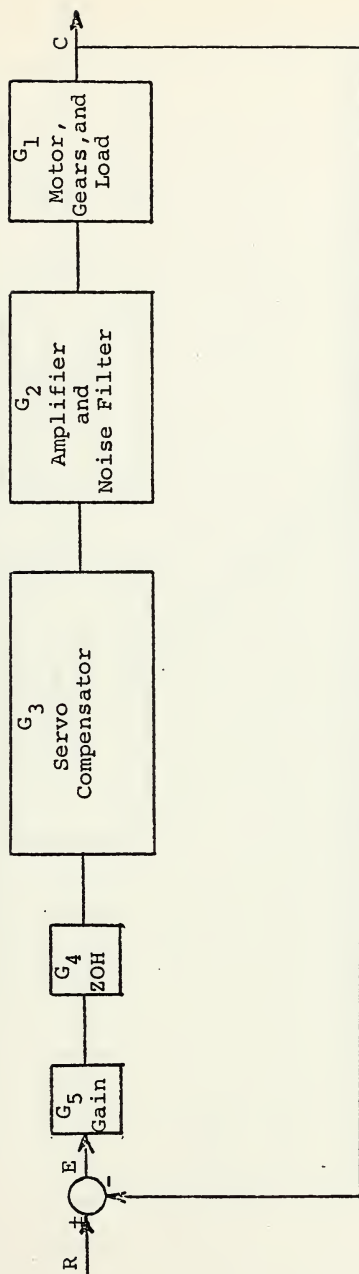


## 5. Compensator Design, Example 5.

This final compensator design example represents a problem of a slightly different nature than those presented to this point. The servo system under consideration here has a zero order hold present in the error channel and in addition, due to the large coefficient values that result in the plant transfer function, the problem requires frequency scaling in order to insure that the numerical limitations of the particular computer that was being used would not be exceeded. The original single loop system is shown in the block diagram of figure IV-7. Series compensation of this system was desired in order to meet the following specifications; 1.) the open loop magnitude at 377 radians/second  $\geq 30\text{db}$ , 2.) the gain crossover frequency  $\geq 1885$  radians/second, and 3.) the phase margin at gain crossover  $\geq 45$  degrees. These specifications were to be met using only a series compensator and the parameters of that part of the system shown in figure IV-7, other than the compensator, was to remain unchanged. An initial attempt at using the CALICO program to design a compensator resulted in a warning of excessive number size during computation and recommended frequency scaling of the problem. Thus a scale factor of 1000 was selected as a convenient value and the entire system, including the sampling frequency of the zero order hold, was scaled down in frequency by the value of this scale factor. Using Laplace transform notation, this was accomplished by a straight-forward substitution of  $s = 1000S$ , where  $S$  represents the new scaled frequency values. A block diagram showing the values of the system parameters after scaling is given in figure IV-7A. When this type of scaling is performed the resultant numerical values of the program need only be multiplied by the appropriate constant value in order to be corrected for use

in the unscaled system.

Originally, work on this particular system indicated that a fifth order compensator could provide the necessary phase margin and gain crossover frequency, but that the 30 db specification at 377 radians/second was not satisfied. Thus, with this information as a starting point the magnitude profile of the system with the fifth order compensator in the loop was adjusted at the low frequency end to satisfy the 30 db requirement at the scaled frequency value of 0.377 radians/second. The phase profile was left unchanged. The results of the program can be seen in figures IV-7E through IV-7I. As can be seen from the graphical output the desired specifications have been satisfied by the resulting compensated system.



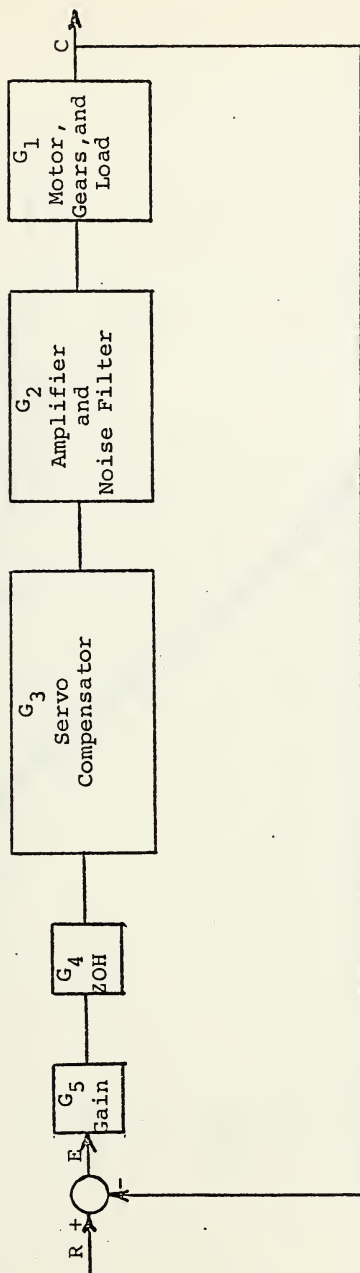
$$ZOH \quad T = -\frac{1}{1560} \text{ seconds}$$

$$G_1 = \frac{558.6}{s(0.0125s + 1)}$$

$$G_2 = \frac{4.51638 \times 10^{12}}{(s^2 + 5.0265 \times 10^3 s + 1.01064 \times 10^8)}$$

$$G_5 = 0.5$$

Unscaled System Block Diagram  
Figure IV-7



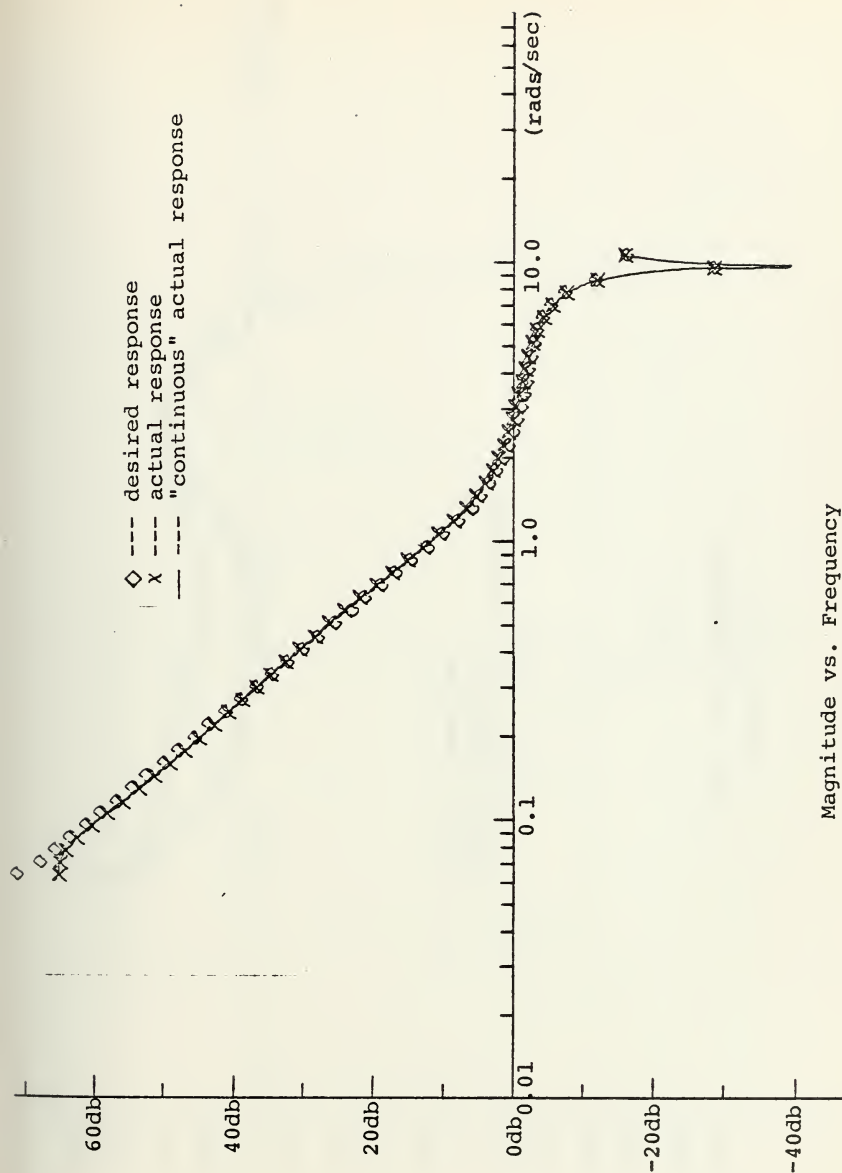
$$G_1 = \frac{4.4688 \times 10^{-1}}{s(s + 8.0 \times 10^{-4})}$$

$$\text{ZOH} \quad T = -\frac{1}{1.560} \text{--seconds}$$

$$G_2 = \frac{8.085179 \times 10^3}{s^2 + 5.02654s + 101.0647}$$

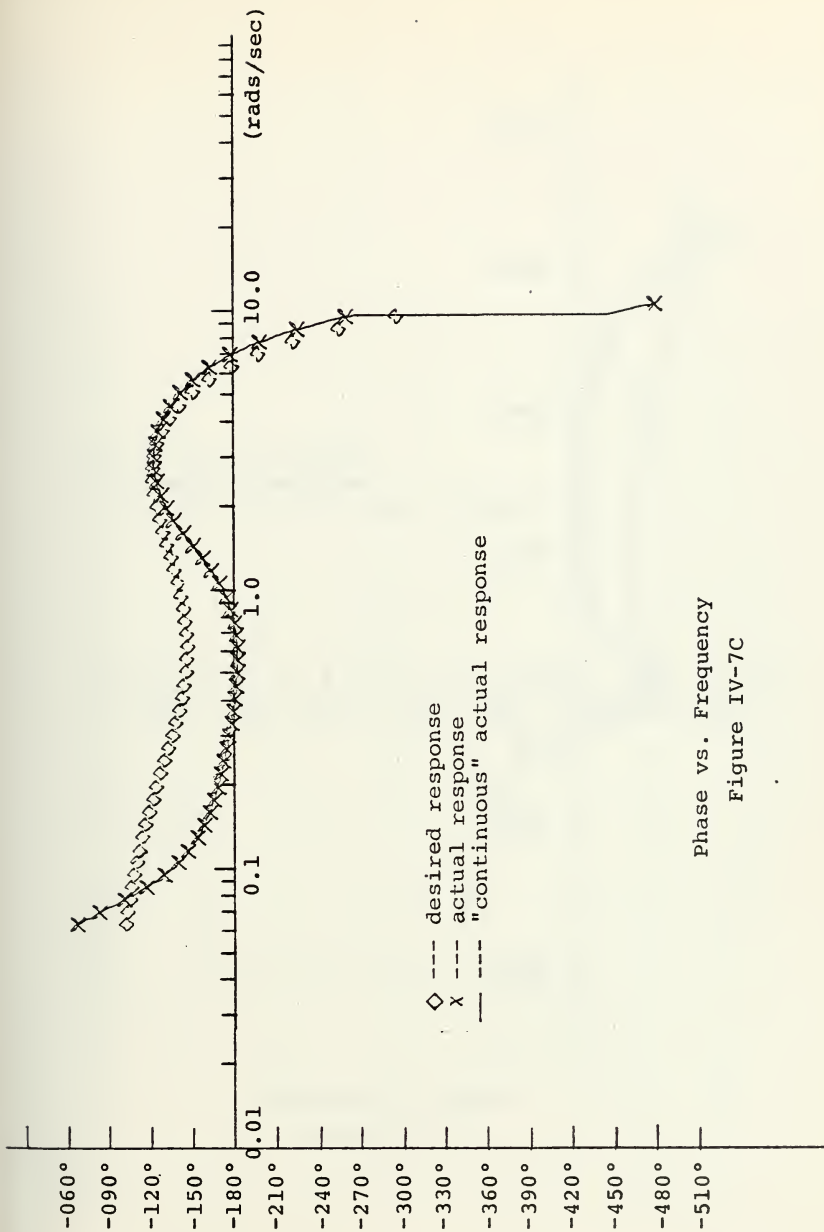
$$G_5 = 0.5$$

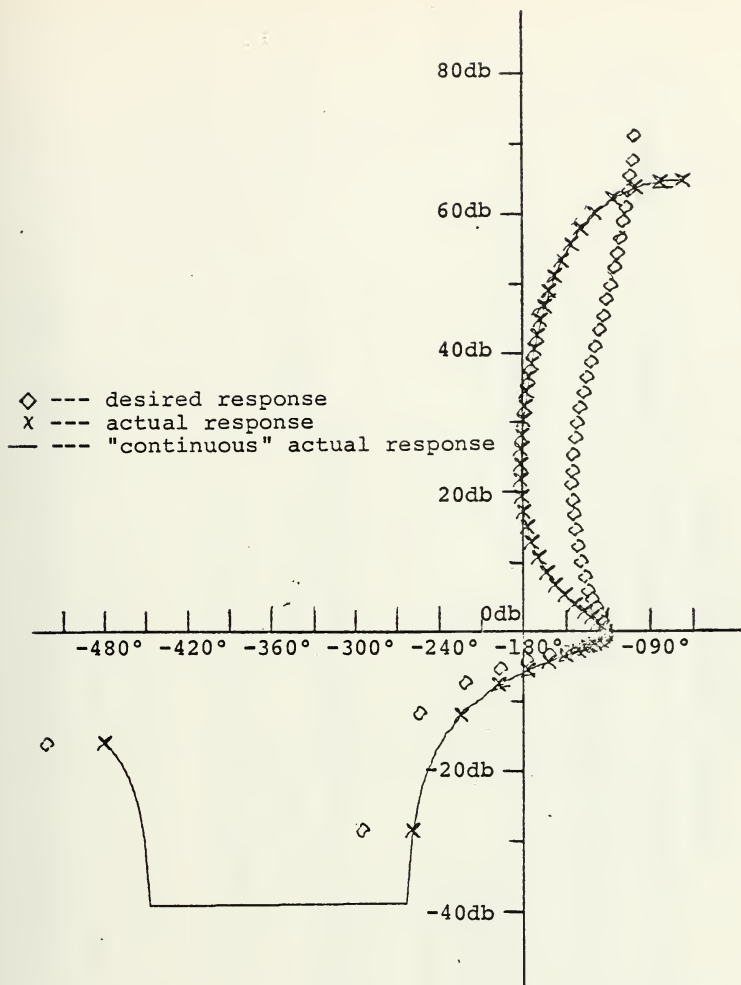
Scaled System Block Diagram  
Figure IV-7A



Magnitude vs. Frequency

Figure IV-7B





Magnitude vs. Phase  
 Figure IV-7D



TITLE --- COMPENSATOR OPTIMIZATION TEST RUN 9 SCALED 1000S

UNCOMPENSATED TRANSFER FUNCTION GAIN = 9.523400E 04

UNCOMPENSATED TRANSFER FUNCTION GAIN  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E-00

UNCOMPENSATED TRANSFER FUNCTION OF LIMITATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

0.0 3.055179E 01 1.050300E 02 5.026546E 00 1.000000E 00

APC: UNCOMPENSATED TRANSFER FUNCTION OF LIMITATOR  
REAL PART IMAGINARY PART

0.0 0.0

-8.000000E-01 0.0

-2.513272E 00 -9.133870E 00

-2.513273E 00 9.133870E 00

COMPENSATOR TRANSFER FUNCTION GAIN = 1.000000E 00

COMPENSATOR TRANSFER FUNCTION GAIN  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 5.000000E 00 1.000000E 01 1.000000E 01 5.000000E 00  
1.000000E 00

Computer Numerical Output  
Figure IV-7E

APR: COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS  
REAL PART IMAGINARY PART

-1.040400E-00 0.0  
-1.000000E 00 0.0  
-1.000000E 00 0.0  
-1.000000E 00 0.0  
-1.000000E-00 0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 05 5.000000E 04 1.000000E 04 1.000000E 03 5.000000E 01  
1.000000E 00

APR: COMPENSATOR TRANSFER FUNCTION DERIVATIVE ROOTS  
REAL PART IMAGINARY PART

-1.000000E 01 0.0  
-1.000000E 01 0.0  
-1.000000E 01 0.0  
-1.000000E 01 0.0  
-1.000000E 01 0.0

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINOR-PIRSE  
TYPE. THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

THE SYSTEM HAS A ZERO ORDER HOLD IN THE LOOP.

THE SAMPLING PERIOD T = 5.410200E-01 (SECS)

THE SAMPLING FREQUENCY WS = 9.501457E 00 (RADIAN/S)

Computer Numerical Output

Figure IV-7F

THE MINIMUM COST FUNCTION VALUE = 2.973144E 00  
 THE ERROR RETURN CODE FROM BOXPLX = 2

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
 COEFFICIENTS IN ASCENDING POWERS OF S

2.144453E-06 3.712168E-00 4.494462E-00 3.195418E-00 8.601346E-01  
 6.835622E-02

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
 REAL PART

-1.223979E 00 0.0

-1.759625E-00 0.0

-8.093903E-01 1.164550E 00

-8.090903E-01 -1.164550E 00

-5.776758E-07 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
 COEFFICIENTS IN ASCENDING POWERS OF S

3.275204E 00 3.1180155E 01 6.154180E 02 3.366127E 01 3.157374E-02  
 1.956992E-03

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
 REAL PART

-2.559161E-02 6.413771E-02

-2.559161E-02 -6.913771E-02

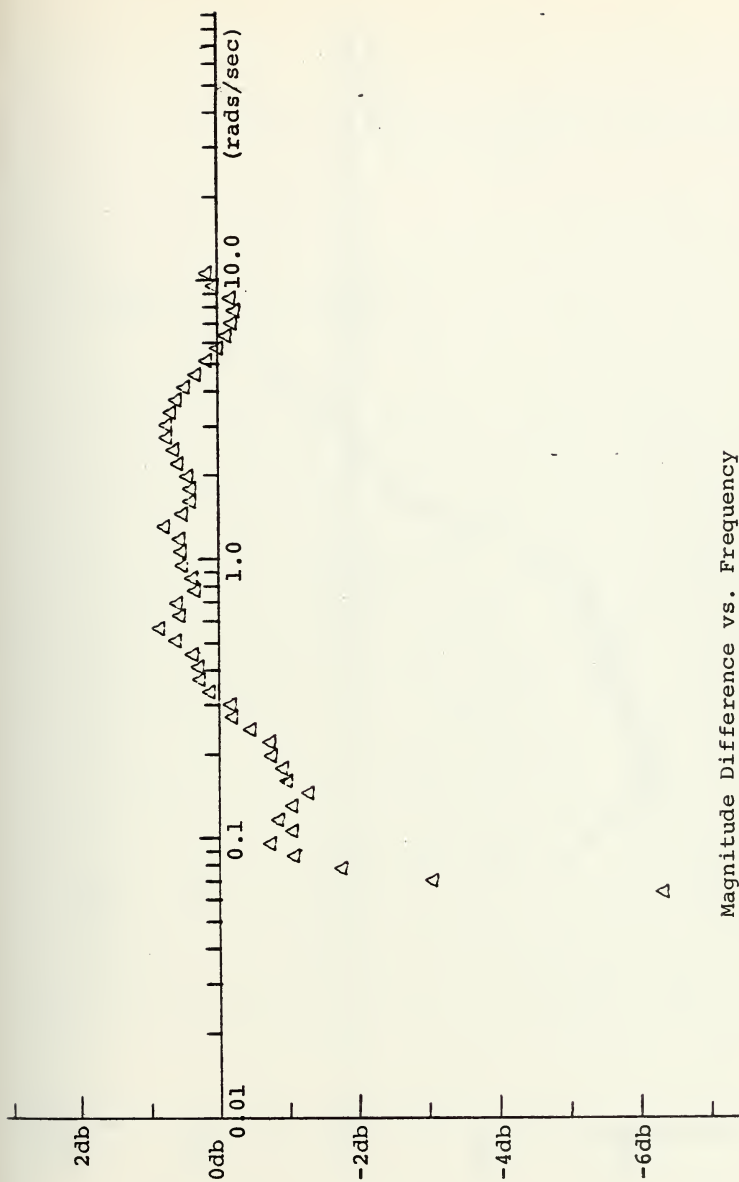
-4.501657E-03 1.403507E-02

-4.501657E-03 -1.403507E 02

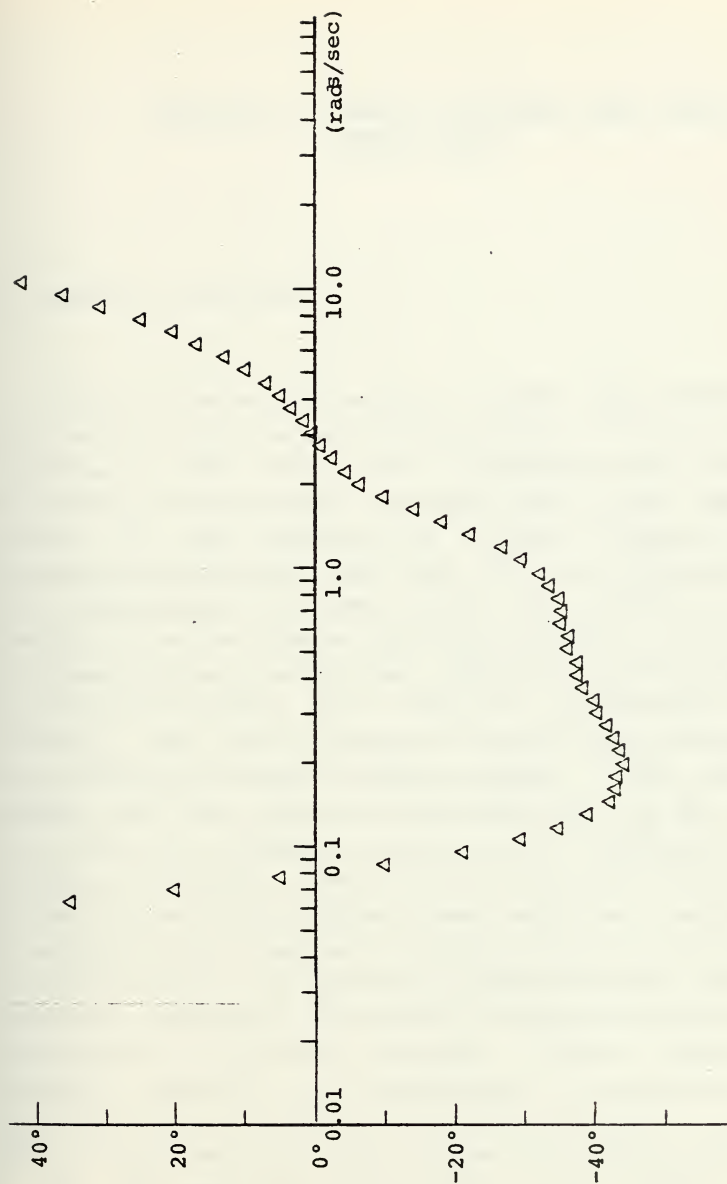
-1.59125E 01 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 3.492921E 01

Computer Numerical Output  
 Figure IV-7G



Magnitude Difference vs. Frequency  
Figure IV-7H



Phase Difference vs. Frequency

Figure IV-7I

## V. SYNTHESIS OF TRANSFER FUNCTIONS FROM FREQUENCY RESPONSE DATA

### A. GENERAL DISCUSSION

In the process of running various test problems to exercise and verify the algorithm presented in the preceding chapters, it was found that in addition to its use as a compensator design program, the basic method could also be applied to the synthesis of transfer functions from frequency response data. That is, if the magnitude and phase measurements of some system are obtained it is possible to use the program (CALICO) in order to obtain an approximate linear analytical expression for the transfer function of the system from which the measurements were obtained. This type of problem is quite often encountered in the modeling process of dynamic systems. That is, the engineer may have measured input-output data of the system in the form of a frequency response and is confronted with obtaining a transfer function which will model this system. This may be the result of the process being too complex to be analyzed and the pertinent equations written on the basis of physical laws or the governing physical laws may not be completely understood in the case of systems involving new technologies or research efforts. Quite often the procedure followed in synthesizing or determining an approximate linear transfer function from frequency response data is to construct a Bode plot for the measured data and to fit the measured characteristic with straight line asymptotic approximations where possible and quadratic

factors where sharp peaks in the measured data occur. As in compensator design by classical frequency domain techniques, this procedure for synthesizing transfer functions can involve a considerable amount of time consuming and tedious "cut and try" on the part of the engineer.

The algorithm previously presented is so structured that the use of the program for this type of transfer function synthesis is possible with no modifications to the coding itself. Since the algorithm was not originally developed for the specific purpose of transfer function synthesis there are several precautions that must be taken in order to insure that the transfer function obtained does indeed adequately represent the system from which the measurements were obtained. In particular when using the program in order to synthesize a transfer function, a separate simulation program for time domain response should be used to verify the transfer function obtained. This procedure of checking the time domain response should be followed for any transfer function obtained from frequency response data. The primary concern here is that it is possible to accurately represent the frequency response of some systems, over a finite frequency range, by a transfer function of higher order than the true transfer function, but in some cases the dominant response to standard time domain inputs will be considerably different than that of the actual system. Also, while there is a direct relationship between the time domain and frequency domain responses through the inverse Fourier transform integral, this integration, strictly speaking, is taken over the entire frequency range from  $-\infty$  to  $+\infty$ . In practical applications it is obviously necessary to limit the range of frequencies to be measured to some finite region. If, however, this region is not sufficiently large for a particular system the resulting transfer function obtained from the frequency response data may not accurately describe the system. If this is the



case, it should become readily apparent in a time domain simulation of the equation obtained from the frequency response data. The possible pitfalls in the use of this algorithm to accomplish transfer function synthesis, such as the one just mentioned, will be discussed later when some example problems will also be presented.

A linear time invariant system may be expressed as a ratio of two frequency dependent polynomials of the form

$$H(s) = \frac{a_n s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0}{b_m s^m + b_{m-1} s^{m-1} + \dots + b_1 s + b_0} = \frac{N(s)}{D(s)}$$

where  $s$  is the frequency dependent complex variable of the Laplace transform. Thus the value of this transfer function is dependent upon the value of  $s$  and the unknown coefficients of the numerator and denominator. Note that this is analogous to the situation that occurs in the design of a cascade compensator for a unity feedback system with the plant represented by a unity gain block. Thus for the purposes of using the program, if we imagine the measured frequency response data to represent some desired system's open loop frequency response profile, and represent the plant of that system by a constant gain of unity, then the transfer function  $H(s)$  to be synthesized may be considered the same as a cascade compensator in the system. Thus by representing the problem in this manner the CALICO program may be used to approximate the transfer function of the actual system by varying the parameters of this pseudo compensator (actually the transfer function of the system) in order to minimize the cost function. The cost function, here represents the error between the measured frequency response data (entered as the desired profile) and the frequency response simulation calculated by the program.

Thus in effect the program minimizes the error between the transfer function approximating the system and the measured frequency response of the system at the discrete frequency points which form the profile input into the program.

As is the case when using the program to design compensators the user must assume a specific order for the numerator and denominator representing, in this case, the transfer function to be synthesized. While unfortunately the program will not vary the order of the numerator and denominator polynomials in order to obtain the best match to the measured frequency response, the user can employ the measured data in order to make an initial guess, as it were, of the form of the transfer function. After this, one may scrutinize the results from the program, which graphs the measured and simulated responses, to make a decision if the form of the transfer function should be adjusted in order to more closely represent the system. Here also, a time domain simulation might be extremely useful in guiding decisions regarding the particular form or modification that should be tried in order to obtain a "good" model for the system under investigation. The fact that the user selects the specific orders of the transfer function numerator and denominator and the minimization routine varies the parameters to obtain a minimum cost function while leaving the orders unchanged presents the potential of obtaining a low order model of a higher order system. While for linear systems, with known transfer functions, there are numerous analytical techniques available for accomplishing this, the capability exists here of essentially obtaining a reduced order transfer function without a priori knowledge of the higher order transfer function describing the system. The only data required by the program is the frequency response of the higher order system and nothing concerning the parameters of the high order transfer function is required.

Since some transfer functions may contain roots of the numerator polynomial in the right half of the s-plane (nonminimum phase systems) this type of transfer function should not be overlooked when attempting to model a system. To this end a nonminimum phase option may be chosen in the CALICO computer algorithm. When this option is selected the search area for the numerator coefficients is expanded to include both positive and negative values and the implicit constraints requiring left half plane roots for the numerator polynomial are eliminated. Selection of this option does not guarantee that a nonminimum phase transfer function will result from the CALICO program but it does allow this type of transfer function to be considered as a feasible solution of the problem.

#### B. TRANSFER FUNCTION SYNTHESIS: EXAMPLE PROBLEMS

In this section several example problems are presented to illustrate the use of the CALICO program in determining transfer functions from frequency response data. These examples illustrate the validity of the basic concept of using the minimization algorithm to determine the transfer function, but they are by no means an exhaustive presentation of all possible variations of cost functions and frequency ranges that may be employed in determining a transfer function representation of a system. For example, as will be shown in the example problems, different transfer function representations may be obtained by considering different ranges of the measured frequency response. That is, while the order of the numerator and denominator are fixed by the user, the parameter values returned from the program may vary depending upon the frequency range being considered. This may be desirable if separate low frequency and high frequency representations are being determined.

Also in using the type one cost function, which includes both phase and magnitude, the cost surface is different than if the magnitude difference alone were used in the cost function. This has an effect on the resulting parameter values, particularly when lower order transfer functions are chosen to represent higher order systems.

The necessary data and the input format employed are almost identical to those used when the program functions in its originally designed compensator design mode. Figure V-1 illustrates the input data deck used for the first synthesis example problem. As can be seen the only significant change is that the plant has been represented by a unity gain transfer function. This has the effect that the resulting "compensator" represents the transfer function modeling the entire system.

(COLUMN NUMBERS, READ VERTICALLY)

TRANSFER FUNCTION SYNTHESIS EXAMP 1 20PT

[illegible][illegible]

131

## 1. Synthesis Example 1.

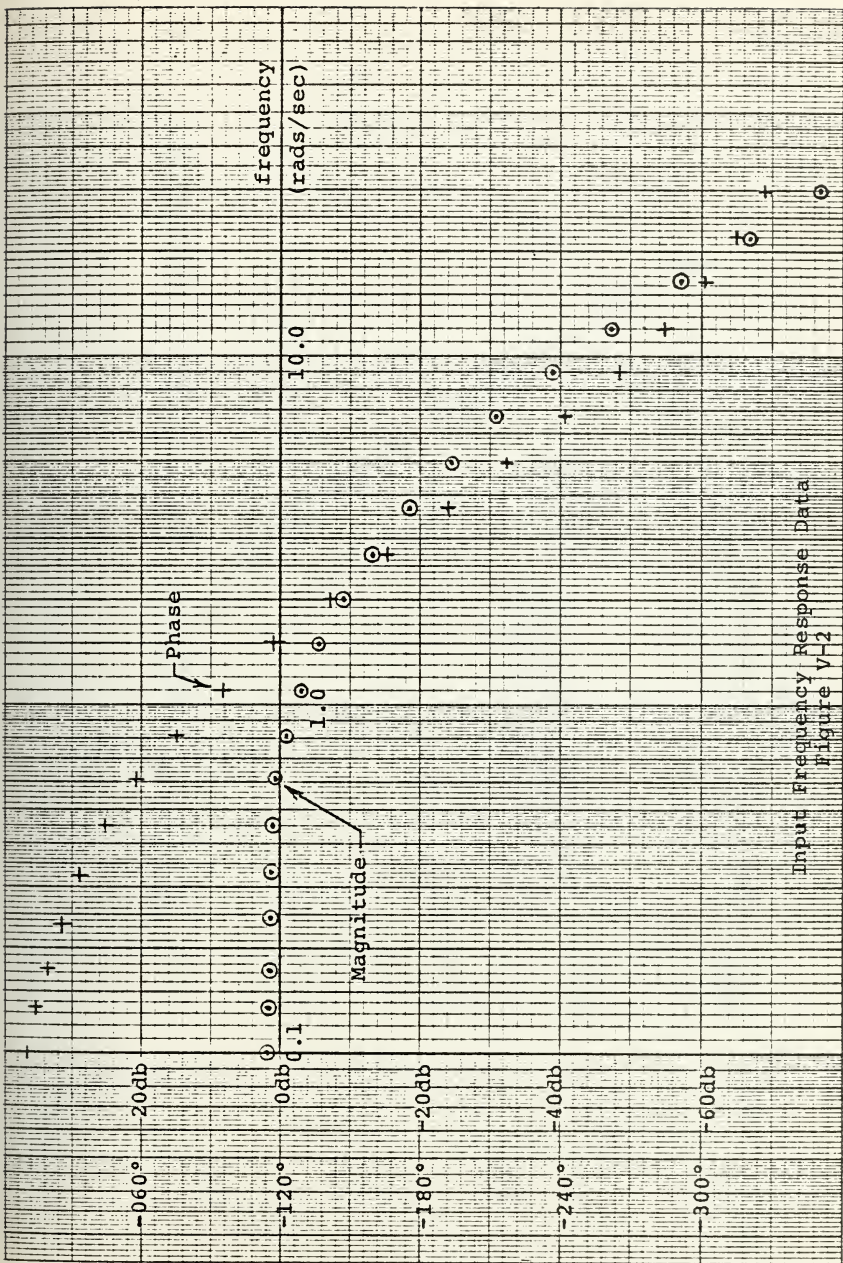
The frequency response data, shown graphically in figure V-2, was generated from a known transfer function using a computer frequency response program. As can be seen, twenty magnitude and phase values were selected at discrete frequencies over the range from 0.1 radians/second to 30 radians/second. Examination of the frequency response data suggests a transfer function of the form of a constant gain over a polynomial of some unknown order. This form will yield a constant magnitude at low frequencies and a decreasing magnitude as the frequency increases. Closer examination of the magnitude curve in the range between 2 and 20 radians/second indicates a slope of approximately 60 db/decade, implying a polynomial of at least third order in the denominator. The phase curve extends from 10 degrees to 330 degrees over the range of frequencies considered. This indicates that the denominator may be of fourth order or higher. As a first guess a third order denominator is assumed. The results from the CALICO program are shown in figure V-2A through V-2H. The initial guess for the denominator roots and the system gain were chosen as 1, as can be seen in figure V-2F. Figures V-2A and V-2D show that the resulting magnitude curve approximates the measured response well at the lower frequencies with a slight deviation beginning to appear at the higher frequencies. The phase plot of figure V-2B and the plot of the difference between the actual phase and desired phase shown in figure V-2E indicate a significant amount of deviation from the measured values in the higher frequency region. This suggests that another pole in the transfer function is necessary to more accurately model the system. Thus, the program was again executed, this time using a fourth order denominator. The results illustrated in figures V-2I



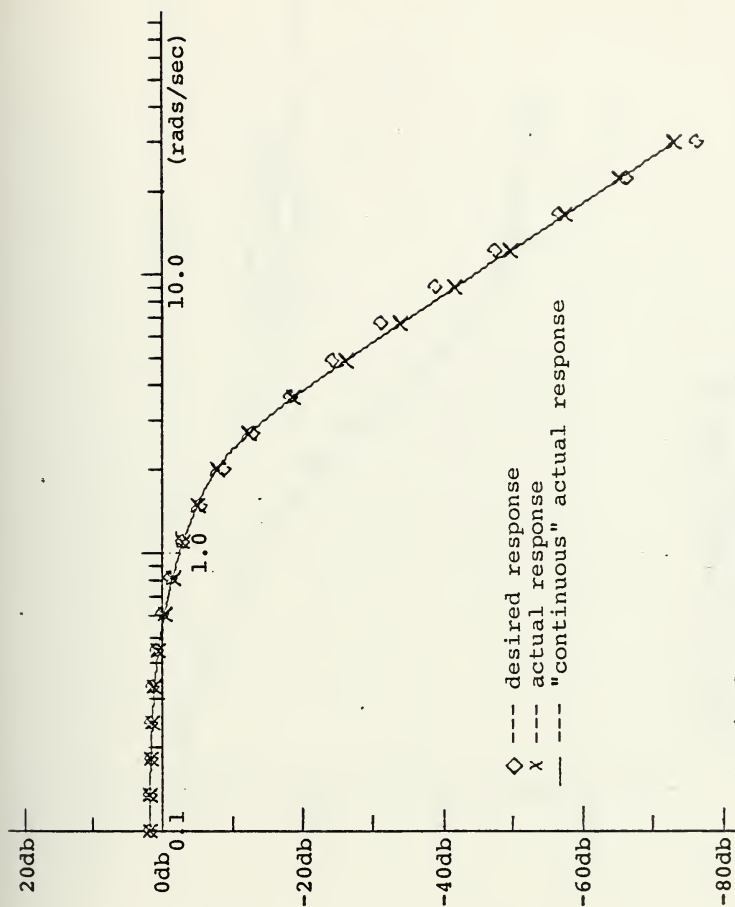
through V-2F indicate that this second form of the transfer function more accurately represents the system. Since the original transfer function is known in this case, an opportunity exists to evaluate the performance of the program in terms of the accuracy of the transfer function returned after minimization of the cost function based on the input frequency response data. This of course would generally not be the case when trying to determine the transfer function of an actual system that was to be modeled. The parameter values returned from the CALICO program shown in figure V-20 may be compared with the original transfer function given below.

$$T(s) = \frac{126.0}{(s + 1)(s + 2)(s + 5)(s + 10)}$$

As can be seen, the values of the transfer function roots returned from the program for the fourth order denominator are in agreement with the values of the original transfer function used to generate the frequency response data. If for the system under consideration only the magnitude response were of interest, the transfer function consisting of a constant gain over a third order denominator might be of sufficient accuracy to model the system. This judgement, of course, rests with the user who is familiar with the intended use of the transfer function modeling the system.

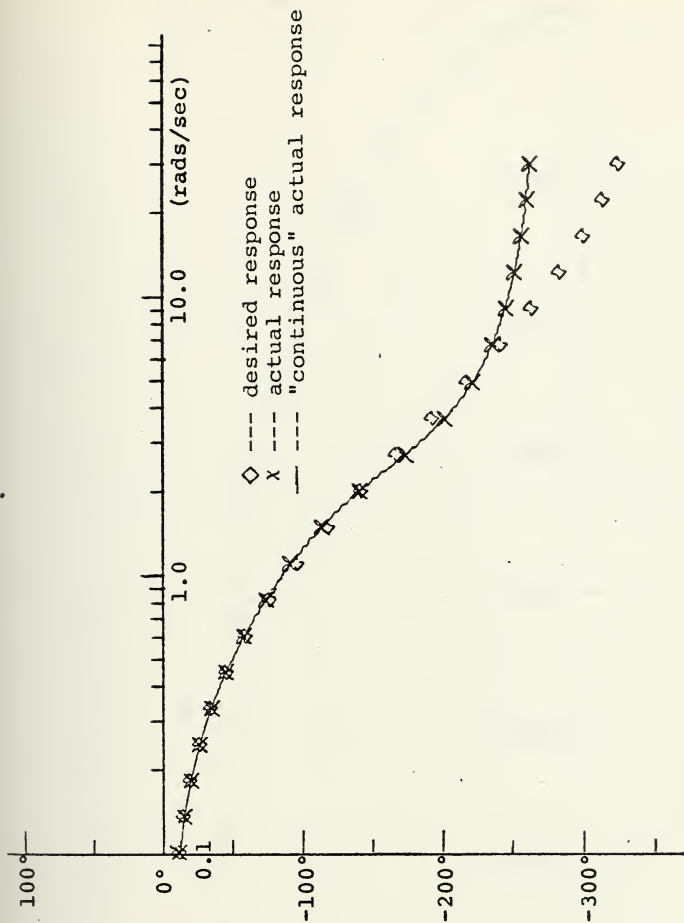






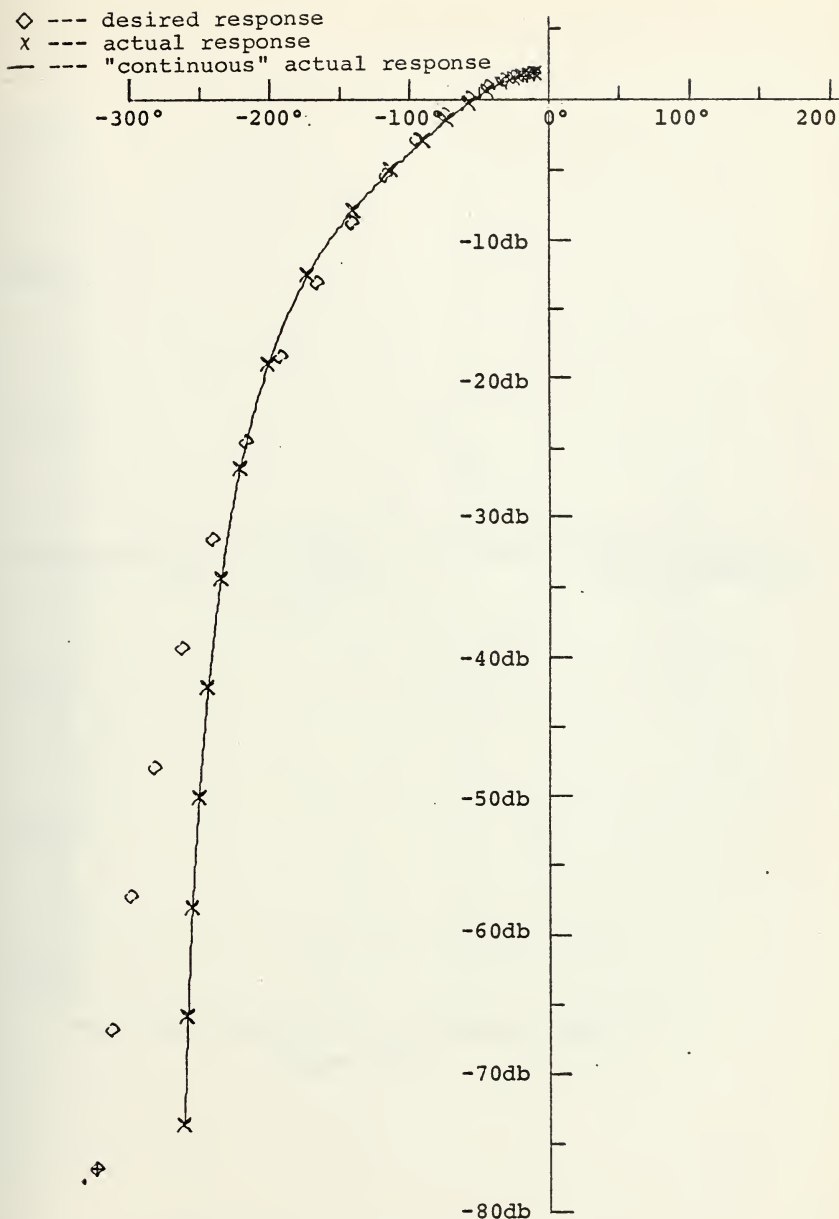
Magnitude vs. Frequency, Run #1

Figure V-2A

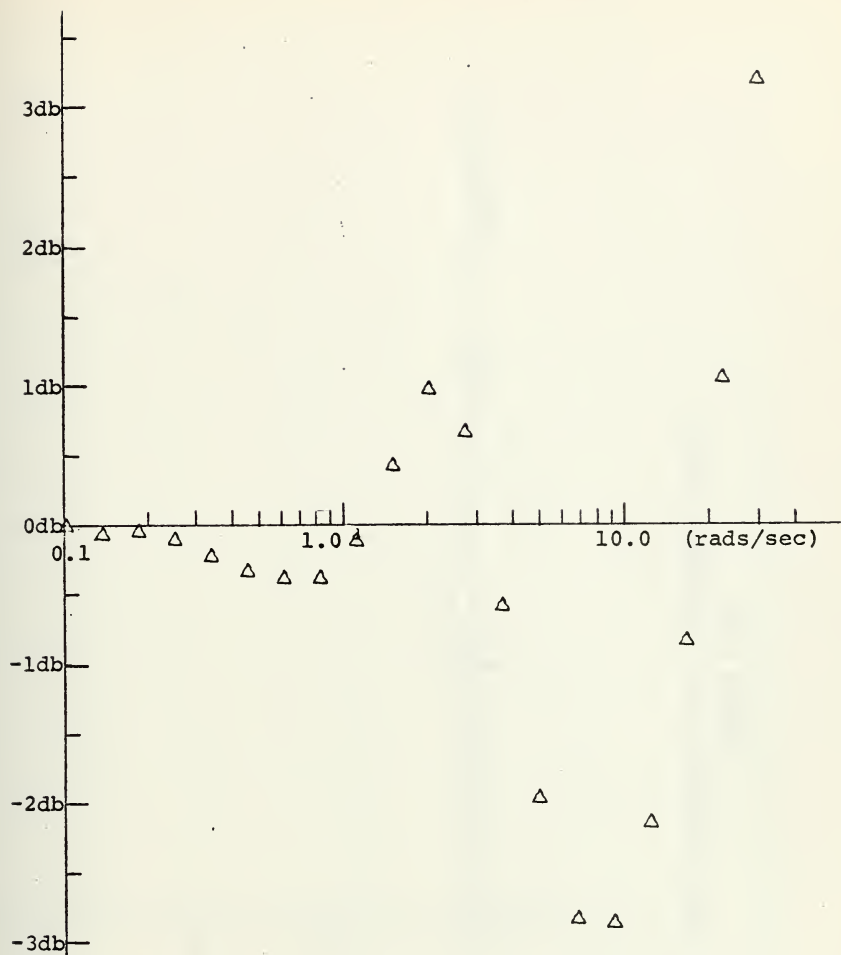


Phase vs. Frequency, Run #1

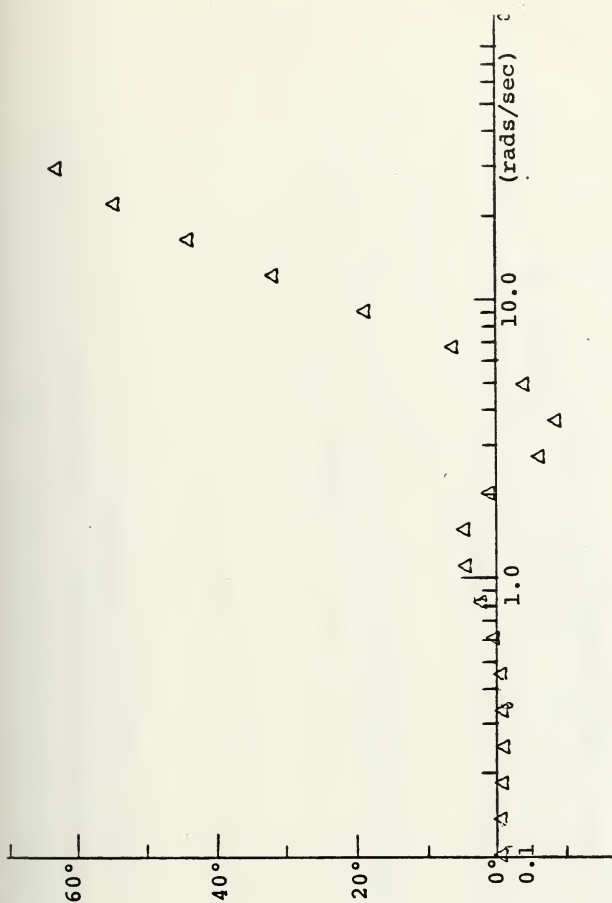
Figure V-2B



Magnitude vs. Phase, Run #1  
Figure V-2C



Magnitude Difference vs. Frequency, Run #1  
Figure V-2D



Phase Difference vs. Frequency, Run #1  
Figure V-2E

TITLE --- TRANSFER FUNCTION SYNTHESIS EXAMP 1 20PT

UNCOMPENSATED TRANSFER FUNCTION GAIN = 1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

COMPENSATOR TRANSFER FUNCTION GAIN = 1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 3.000000E 00 3.000000E 00 1.000000E 00

ARE: COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS  
REAL PART IMAGINARY PART

-1.000000E 00 0.0

-1.000000E 00 0.0

-1.000000E 00 0.0

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE  
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

Computer Numerical Output, Run #1

Figure V-2F

THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

THE COST FUNCTION TO BE USED IS THE TYPE 1

THE MINIMUM COST FUNCTION VALUE = 6.255069E-01  
THE ERROR RETURN CODE FROM BOXPLX = 2

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF  $s$

2.0057495 05

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF  $s$

1.590744E 05	3.000009E 05	1.352915E 05	3.551281E 04
--------------	--------------	--------------	--------------

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE:                      IMAGINARY PART  
   REAL PART

-1.544968E 00 1.953757E 00

-1.544968E 00      -1.958797E 00

-7.1971195-01 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION = 5.647960E 00

Computer Numerical Output, Run #1

Figure V-2G

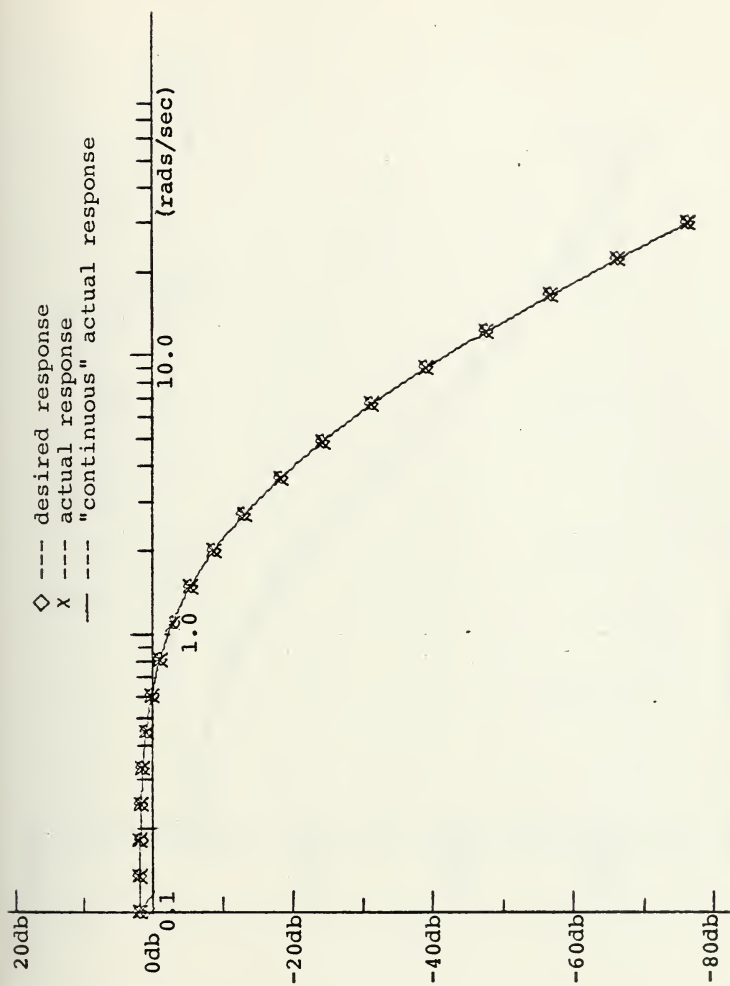
FREQUENCY	MAGNITUDE (DB)	DESIRE D MAG (DB)	PHASE (DEG)	DESIRE D PHASE
9.339956E-02	1.933724	1.938200	-1.073708	-1.030000
1.350000E-01	1.865240	1.938200	-1.446946	-1.390000
1.820000E-01	1.754577	1.754577	-1.538194	-1.870000
2.400000E-01	1.552566	1.655700	-2.590143	-2.509999
3.320000E-01	1.210244	1.437634	-3.423914	-3.350000
4.489999E-01	6.472922	9.843595	-4.531733	-4.450000
6.059999E-01	-2.082647	1.719952	-5.782793	-5.839999
8.130000E-01	-1.417422	1.031741	-7.312377	-7.550000
1.099999E-00	-2.503166	-2.865421	-9.093766	-9.550000
1.490000E-00	-5.045173	-5.465456	-1.132375	-1.180000
2.005559E-00	-7.919118	-8.898106	-1.409275	-1.420000
3.715959E-00	-1.252182	1.319112	-1.731349	-1.670000
5.669999E-00	-1.855828	1.841637	-2.014789	-1.930000
6.650000E-00	-2.546900	3.452326	-2.213355	-2.180000
9.020000E-00	-3.431555	3.150235	-2.356471	-2.420000
1.220000E-01	-4.218900	3.933153	-2.511523	-2.640000
1.650000E-01	-5.031122	4.743710	-2.581773	-2.840000
2.220000E-01	-5.797523	5.713569	-2.566531	-3.010000
3.000000E-01	-6.572250	5.678268	-2.601911	-3.150000
3.000000E-01	-7.357813	-7.677264	-2.627043	-3.260000

THE RECUH TEST OF THE CHARACTERISTIC EQUATION INDICATES  
THAT THE SYSTEM IS STABLE

Computer Numerical Output, Run #1

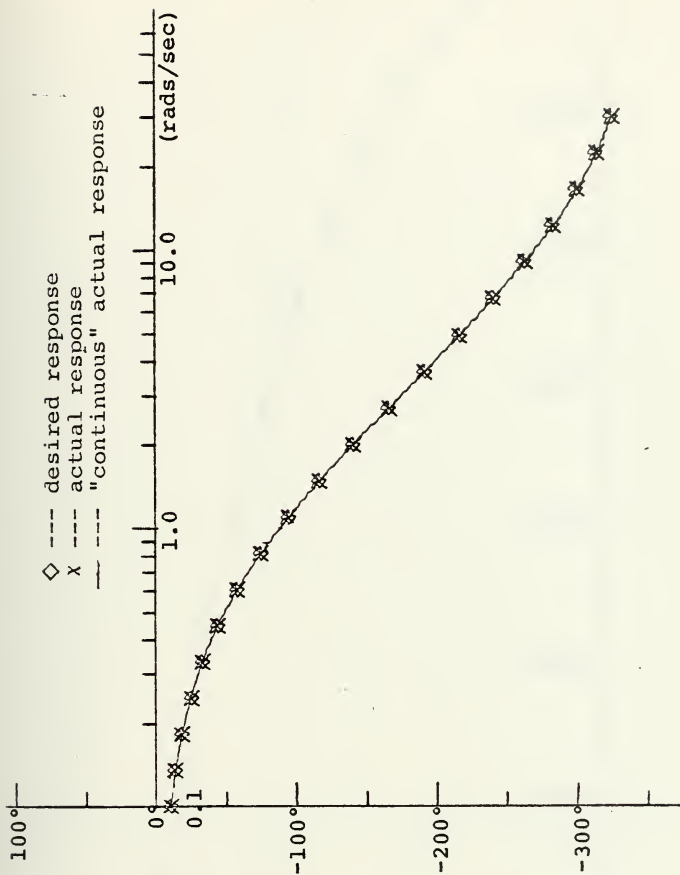
Figure V-2H



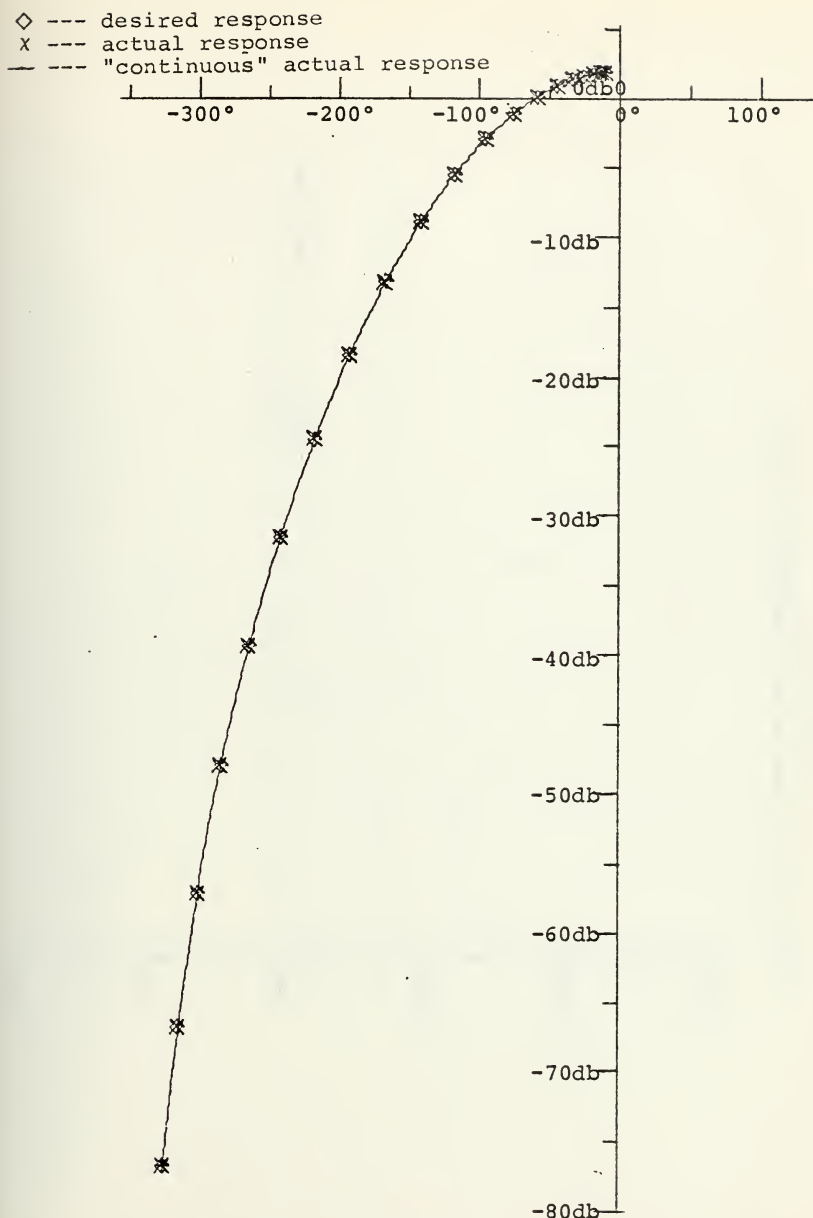


Magnitude vs. Frequency, Run #2

Figure V-2I

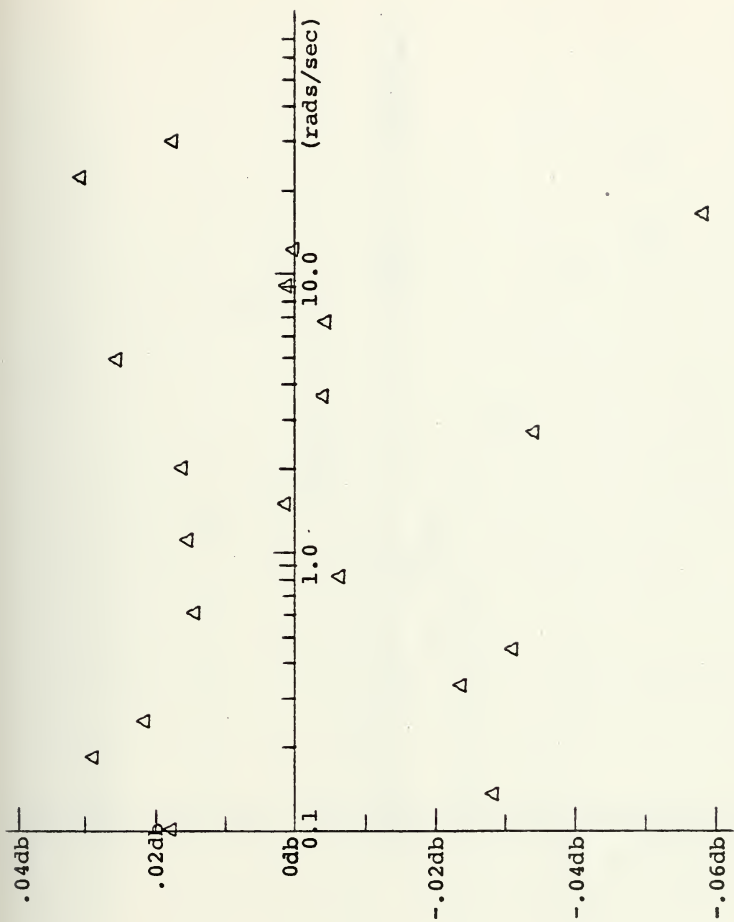


Phase vs. Frequency, Run #2  
Figure V-2J

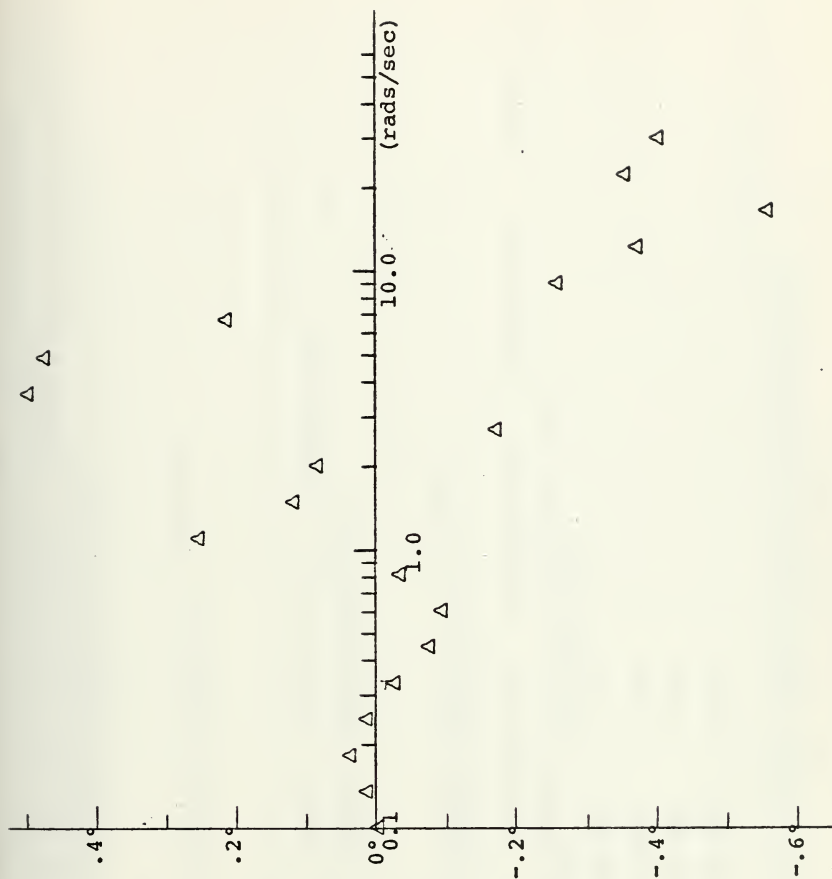


Magnitude vs. Phase, Run #2

Figure V-2K



Magnitude Difference vs. Frequency, Run #2  
Figure V-2L



Phase Difference vs. Frequency, Run #2

Figure V-2M

TITLE --- TRANSFER FUNCTION SYNTHESIS EXAMP 1 2DPT

UNCOMPENSATED TRANSFER FUNCTION GAIN = 1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

COMPENSATOR TRANSFER FUNCTION GAIN = 1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 4.000000E 00 6.000000E 00 4.000000E 00 1.000000E 00

ARC: COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS  
REAL PART IMAGINARY PART

-1.000000E 00 0.0

-1.000000E 00 0.0

-1.000000E 00 0.0

-1.000000E 00 0.0

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE  
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

Computer Numerical Output, Run #2

Figure V-2N

THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

THE COST FUNCTION TO BE USED IS THE TYPE 1

THE MINIMUM COST FUNCTION VALUE = 1.911033E-04

THE ERROR RETURN CODE=FFCM EXPLX = -1

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

7.718555E 00

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

6.122410E 00 1.103352E 01 5.943789E 00 1.100959E 00 6.122756E-02

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE: REAL PART IMAGINARY PART

-9.931583E 00 0.0

-5.054280E 00 0.0

-1.998310E 00 0.0

-9.968116E-01 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 1.260640E 02

Computer Numerical Output, Run #2

Figure V-20

FREQUENCY	MAGNITUDE (DB)	DESIRED MAG (DB)	PHASE (DEG)	DESIRED PHASE
9.959996E-02	1.553922E 00	1.938200E 00	-1.030384E 01	-1.030000E 01
1.350000E-01	1.505704E 00	1.933200E 00	-1.389632E 01	-1.390000E 01
1.820000E-01	1.236566E 00	1.738090E 00	-1.386612E 01	-1.387000E 01
2.450000E-01	1.672922E 00	1.655700E 00	-2.538612E 01	-2.539000E 01
3.320000E-01	1.412649E 00	1.437634E 00	-3.32655E 01	-3.350000E 01
4.489999E-01	9.534442E-01	9.843595E-01	-4.457687E 01	-4.450000E 01
6.059999E-01	1.826766E-01	1.716952E-01	-5.849570E 01	-5.839999E 01
8.180000E-01	1.038156E 00	-1.031741E 00	-7.553554E 01	-7.550000E 01
1.059999E-00	-2.850209E 00	-2.865421E 00	-9.524660E 01	-9.550300E 01
1.490000E 00	-5.466699E 00	-5.465455E 00	-1.178840E 02	-1.180000E 02
2.009999E 00	-8.319533E 00	-8.319106E 00	-1.419164E 02	-1.420000E 02
2.719999E 00	-1.322533E 01	-1.319112E 01	-1.611723E 02	-1.670000E 02
3.659999E 00	-1.848046E 01	-1.841637E 01	-1.925000E 02	-1.930000E 02
4.950000E 00	-2.445344E 01	-2.452426E 01	-2.175242E 02	-2.180000E 02
6.450000E 00	-3.150651E 01	-3.150235E 01	-2.417667E 02	-2.420000E 02
9.030000E 00	-3.533047E 01	-3.933153E 01	-2.647620E 02	-2.640000E 02
1.220000E 01	-4.753764E 01	-4.793710E 01	-2.843740E 02	-2.840000E 02
1.650000E 01	-5.716837E 01	-5.713869E 01	-3.015610E 02	-3.010000E 02
2.220000E 01	-6.618203E 01	-6.678268E 01	-3.155562E 02	-3.150000E 02
3.000000E 01	-7.615511E 01	-7.677264E 01	-3.264048E 02	-3.260000E 02

THE ROOT TEST OF THE CHARACTERISTIC EQUATION INDICATES  
THAT THE SYSTEM IS STABLE

Computer Numerical Output, Run #2

Figure V-2P

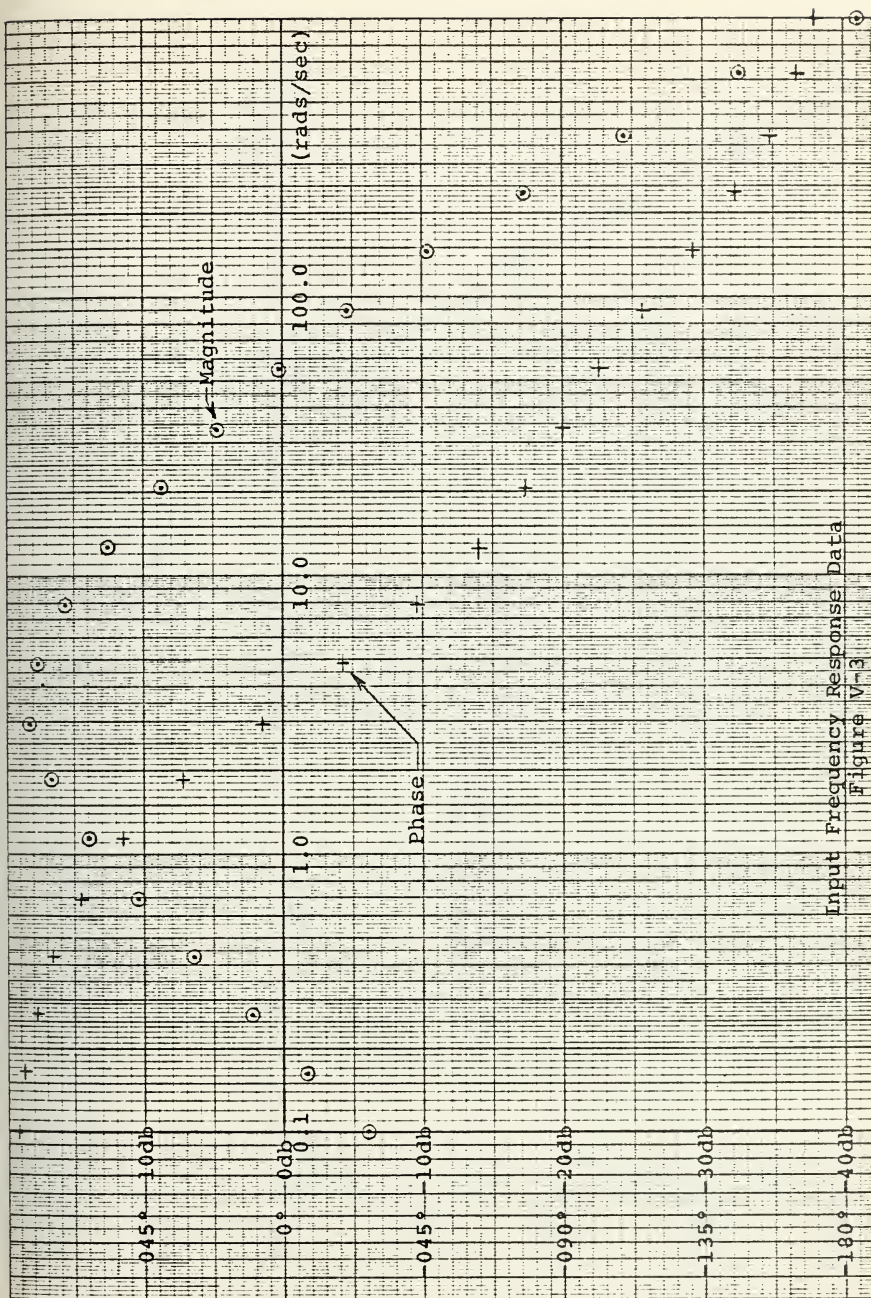


## 2. Synthesis Example 2.

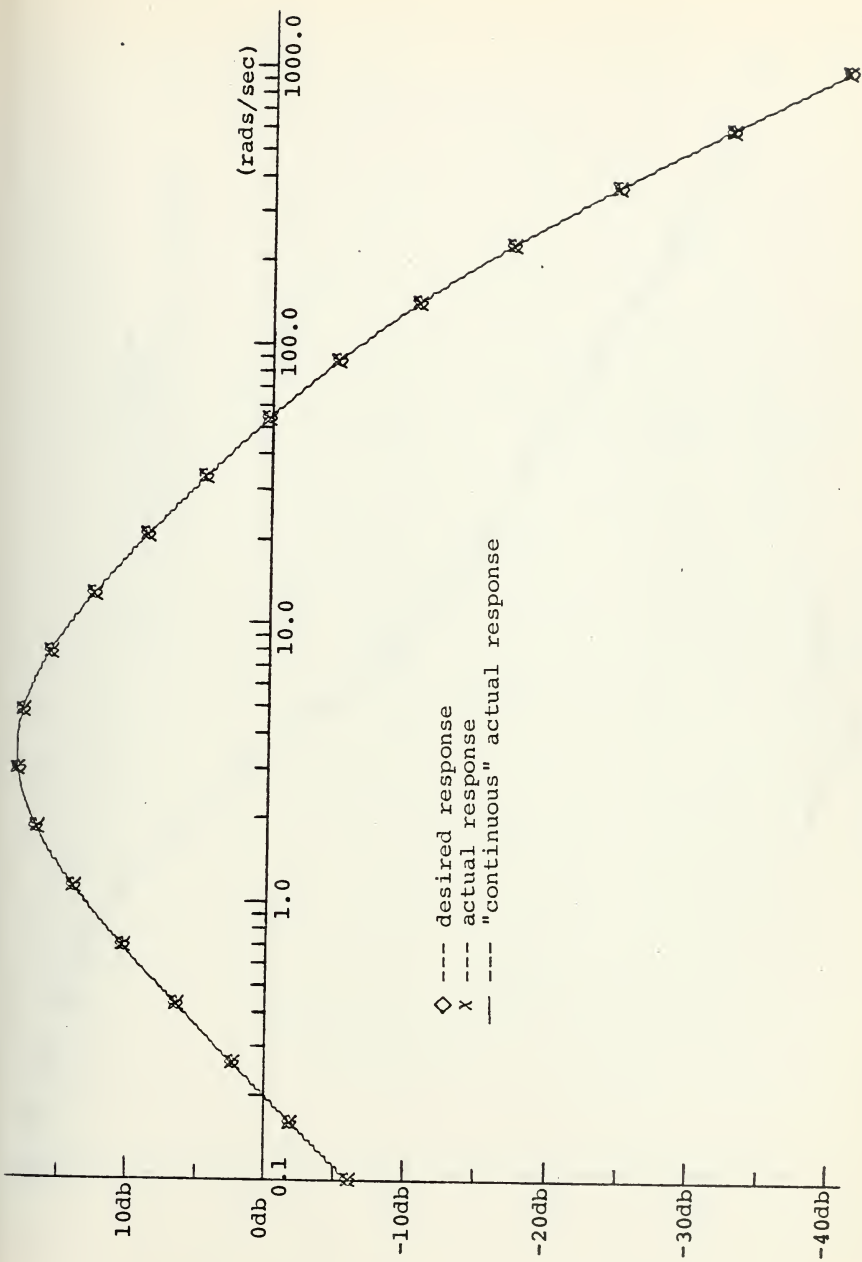
As a second synthesis example, consider the frequency response data shown in figure V-3. Here the magnitude and phase response for a system are shown at twenty distinct frequency values over the range from 0.1 to 1000.0 radians/sec. As can be seen from this figure the positive slope of 20 db/decade of the magnitude curve at the lower frequencies suggests a transfer function with at least one zero. This is also confirmed by the phase response, which at the lower frequencies initially has a value of 90 degrees. At the higher frequencies inspection of the magnitude plot shows a negative slope of approximately 40 db/decade, indicating the dominance of a third order denominator in this area. The phase plot approaching -180 degrees at these higher frequencies also suggests this. Thus, initially a form for the transfer function consisting of a first order numerator and a third order denominator was selected to model the system from which the frequency response data was obtained. The results are shown in figures V-3A through V-3H. As can be seen from the graphical output presented in the figures, this particular form of transfer function appears to match the measured frequency response quite well over the entire range of frequencies considered. The roots and gain of the transfer function returned from the program are shown in figures V-3G and V-3H. As in the previous example it happens that the exact transfer function used to generate the measured data input to the CAIICO program is known. This transfer function is given below in order that the reader may compare the values of the original transfer function parameters with those returned from the program.

$$T(s) = \frac{9000s}{(s + 2.4)(s + 5)(s + 150)}$$

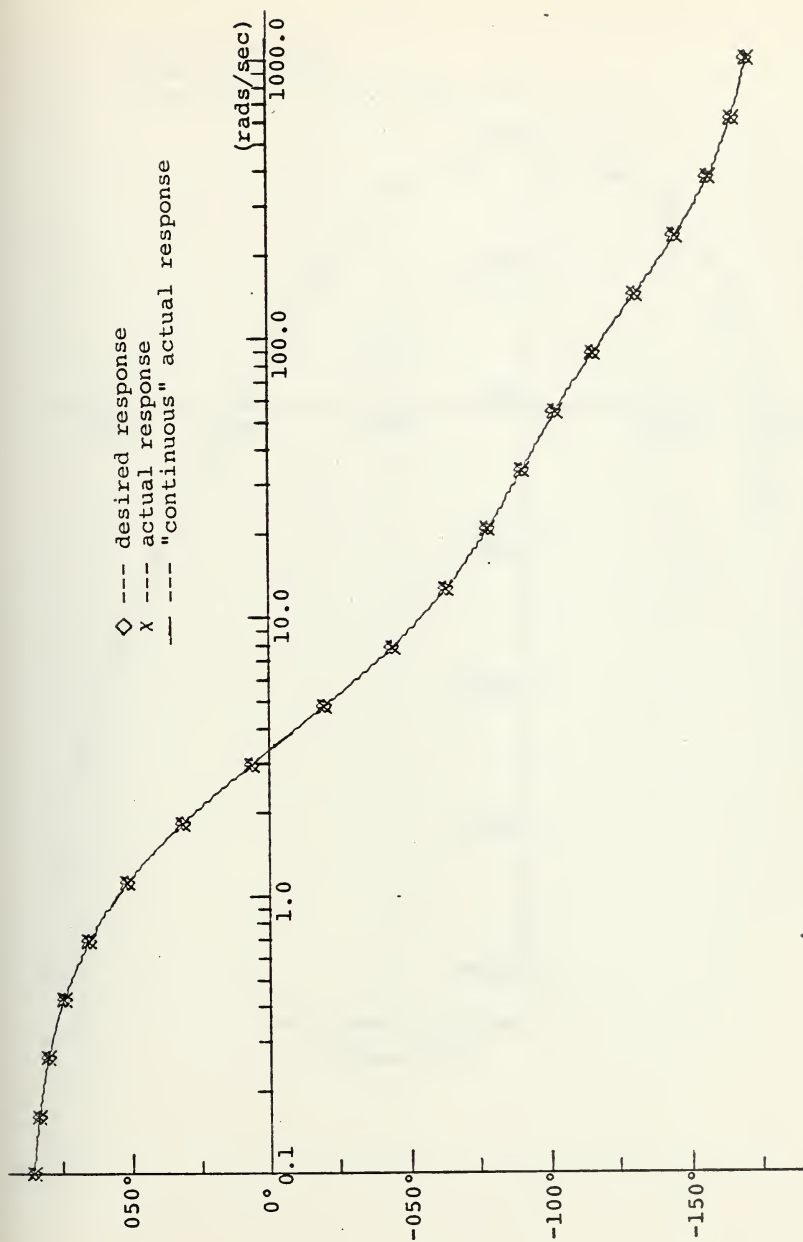
As can be seen from the data presented in figures V-3G and V-3H, the parameter values returned from the program are in close agreement with the original transfer function values. While it is unlikely that they will ever be exactly the same as the original values there is certainly not a significant difference between the two sets of values for most engineering purposes.



Input Frequency Response Data  
Figure V-3

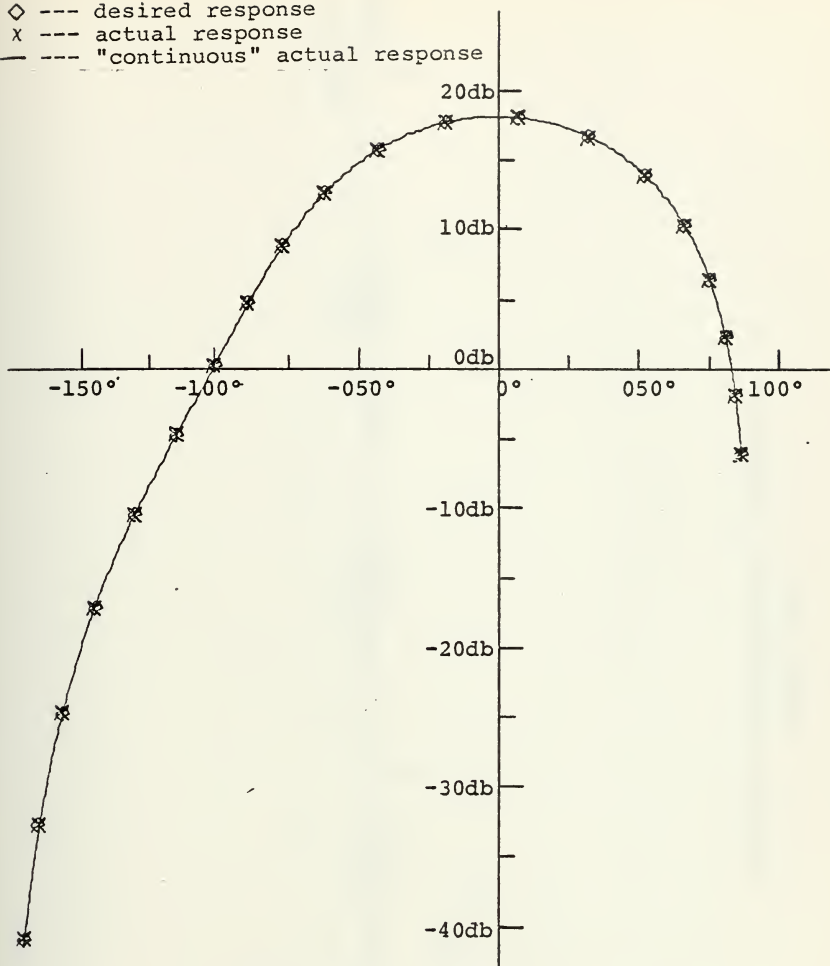


Magnitude vs. Frequency  
Figure V-3A

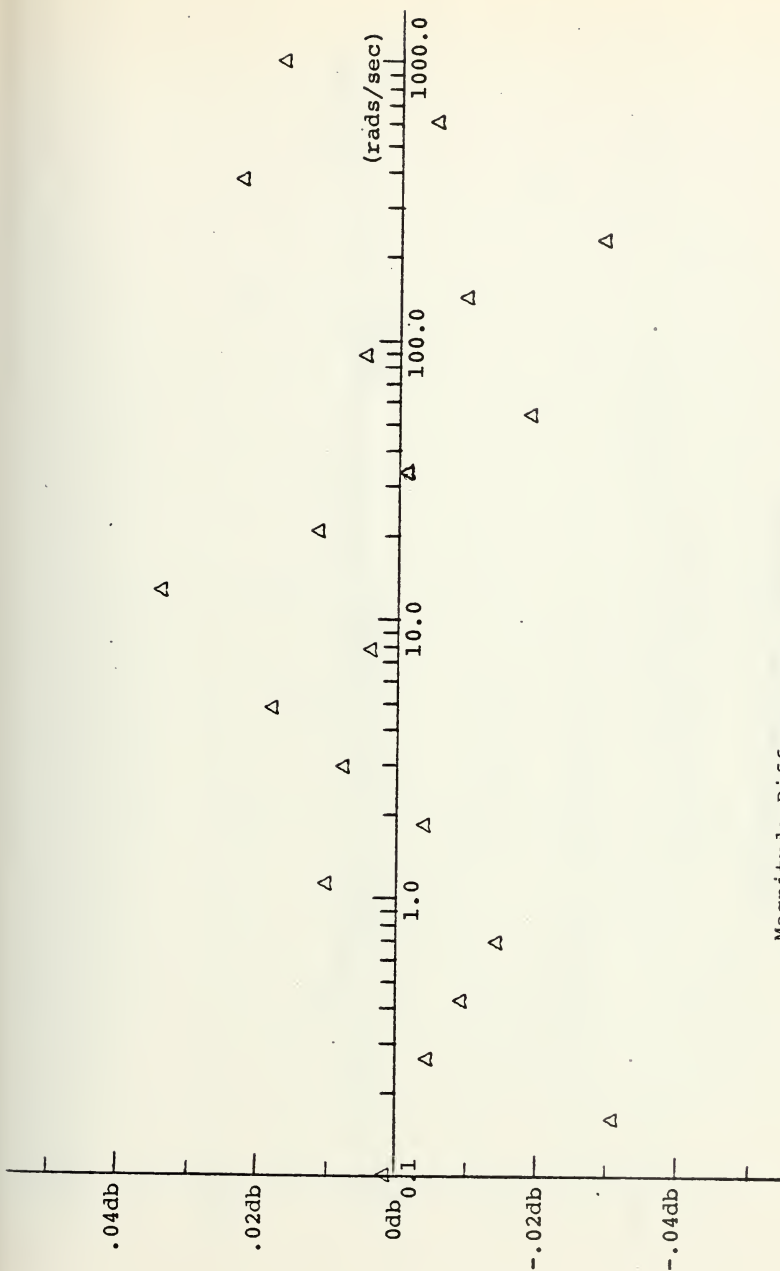


Phase vs. Frequency  
Figure V-3B

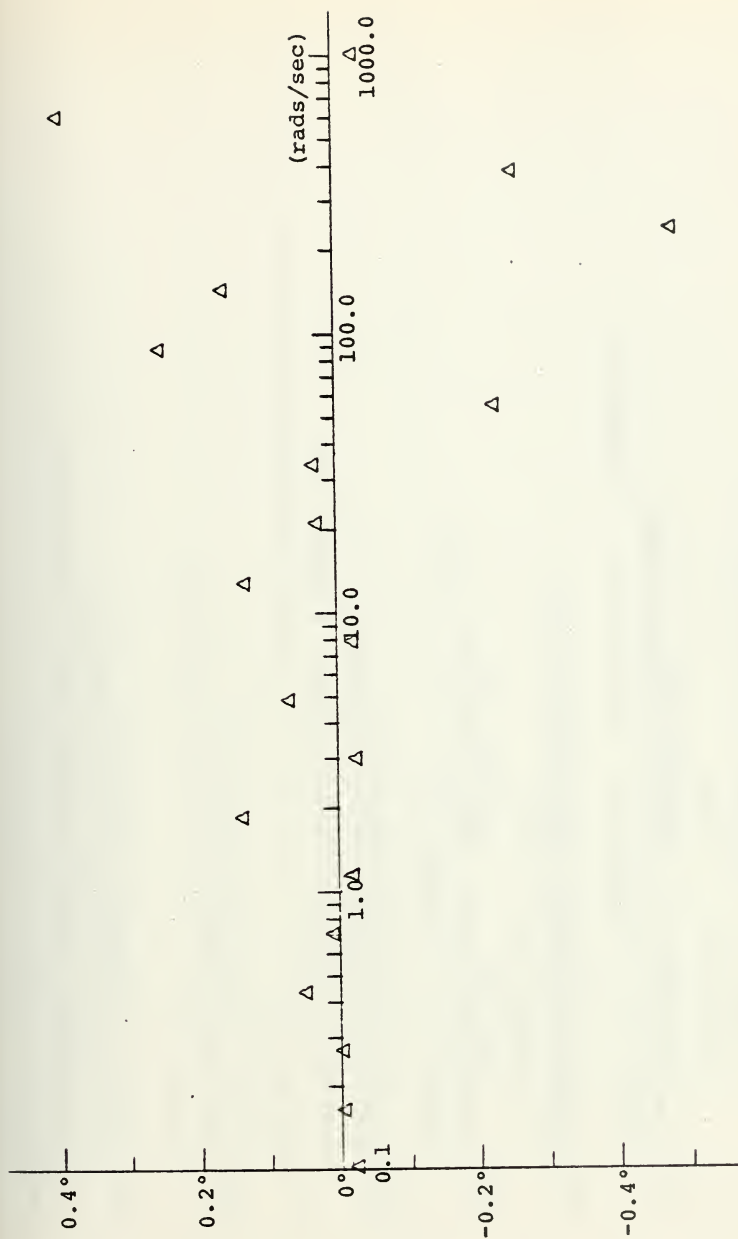
◇ --- desired response  
 x --- actual response  
 — --- "continuous" actual response



Magnitude vs. Phase  
 Figure V-3C



Magnitude Difference vs. Frequency  
Figure V-3D



Phase Difference vs. Frequency  
Figure V-3E



TITLE --- SYNTHESIS EXAMPLE TWO

UNCOMPENSATED TRANSFER FUNCTION GAIN = 1.000000E 00  
UNCOMPENSATED TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

COMPENSATOR TRANSFER FUNCTION GAIN = 1.000000E 00  
COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS  
ARE: REAL PART IMAGINARY PART

-1.000000E 00 0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 3.000000E 00 3.000000E 00 1.000000E 00

COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS  
ARE: REAL PART IMAGINARY PART

-1.000000E 00 0.0

-1.000000E 00 0.0

-1.000000E 00 0.0

Computer Numerical Output

Figure V-3F

-----  
THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE  
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION  
-----

-----  
THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000  
-----

-----  
THE COST FUNCTION TO BE USED IS THE TYPE 1  
-----

-----  
THE MINIMUM COST FUNCTION VALUE = 1.506277E-04  
THE ERROR RETURN CODE FROM BOXPLX = 0  
-----

-----  
OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S  
-----

2.618764E 01    2.443936E 05

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE:

REAL PART                      IMAGINARY PART  
-----  
-1.071535E-04                      0.0  
-----

Computer Numerical Output  
Figure V-3G

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

4.891628E 04 3.042503E 04 4.272676E 03 2.710205E 01

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE:  
REAL PART  
IMAGINARY PART

-1.502601F 02 0.0

-4.57837E 00 0.0

-2.41279C 00 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 9.017527E 03

FREQUENCY	MAGNITUDE (DB)	DESIRE MAG (DB)	PHASE (DEG)	DESIRE PHASE
9.99999E-02	-6.03648E 00	-6.037990E 00	8.637637E 01	8.639999E 01
1.619999E-01	-1.861110E 00	-1.830299E 00	8.419530E 01	8.423000E 01
2.640000E-01	2.340505E 00	2.345422E 00	8.059622E 01	8.059999E 01
4.280000E-01	6.435030E 00	6.444381E 00	7.484970E 01	7.479999E 01
6.950000E-01	1.031313E 01	1.039656E 01	6.570982E 01	6.570000E 01
1.128999E 00	1.355662E 01	1.394458E 01	5.167982E 01	5.170000E 01
1.830000E 00	1.263714E 01	1.670111E 01	3.193719E 01	3.179999E 01
2.980000E 00	1.810199E 01	1.809430E 01	6.953293E 00	6.980000E 00
4.830000E 00	1.777023E 01	1.775233E 01	-1.943172E 01	-1.950000E 01
7.849999E 00	1.515552E 01	1.579161E 01	-4.352361E 01	-4.350000E 01
1.270000E 01	1.268311E 01	1.264916E 01	-6.266949E 01	-6.279999E 01
2.070000E 01	8.861033E 00	8.845580E 00	-7.767279E 01	-7.770000E 01
3.509999E 01	4.705646E 00	4.710566E 00	-9.006964E 01	-9.009999E 01
5.459999E 01	2.379835E-01	2.537425E-01	-1.022297E 02	-1.020000E 02
8.859999E 01	-4.650372E 00	-4.701540E 00	-1.157497E 02	-1.160000E 02
1.440000E 02	-1.043155E 01	-1.042867E 01	-1.308414E 02	-1.310000E 02
2.340000E 02	-1.716870E 01	-1.713969E 01	-1.454845E 02	-1.450000E 02
3.750000E 02	-2.461876E 01	-2.470154E 01	-1.572551E 02	-1.570000E 02
6.160000E 02	-3.273282E 01	-3.272775E 01	-1.655042E 02	-1.660000E 02
1.000000E 03	-4.659533E 01	-4.101213E 01	-1.710311E 02	-1.710000E 02

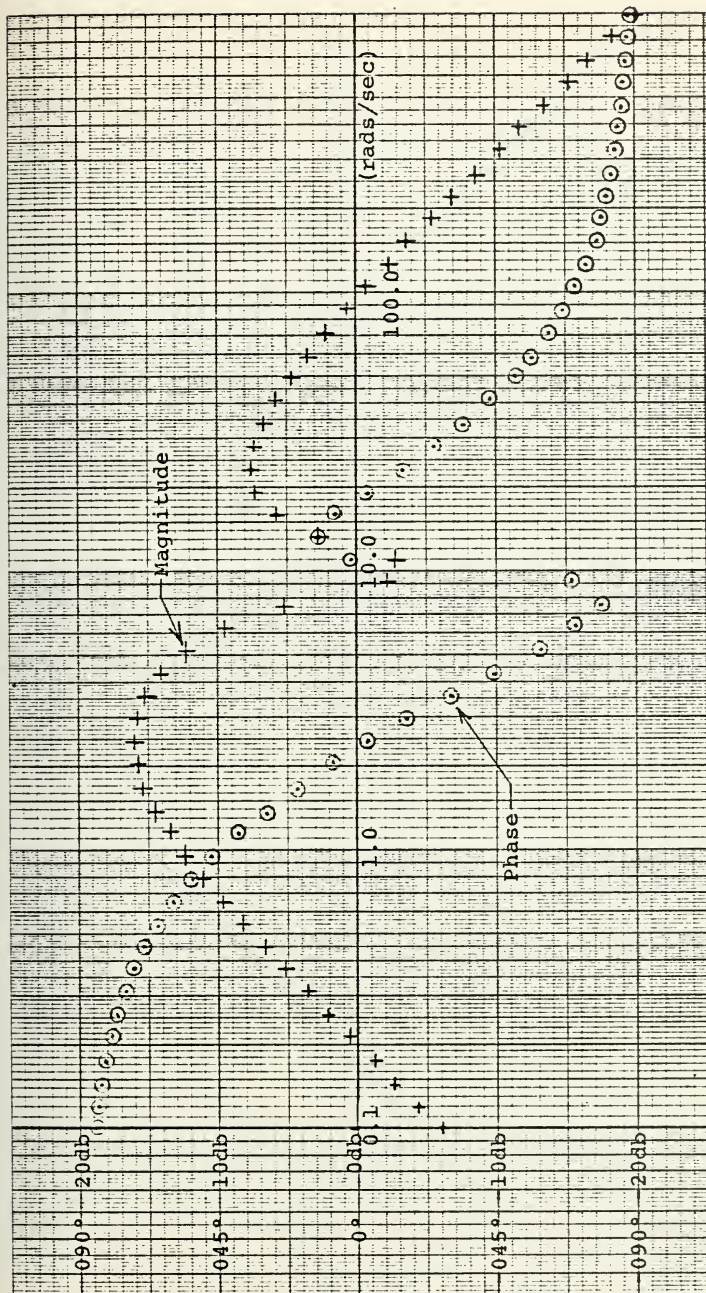
THE ROOTS TEST OF THE CHARACTERISTIC EQUATION INDICATES  
THAT THE SYSTEM IS STABLE

Computer Numerical Output

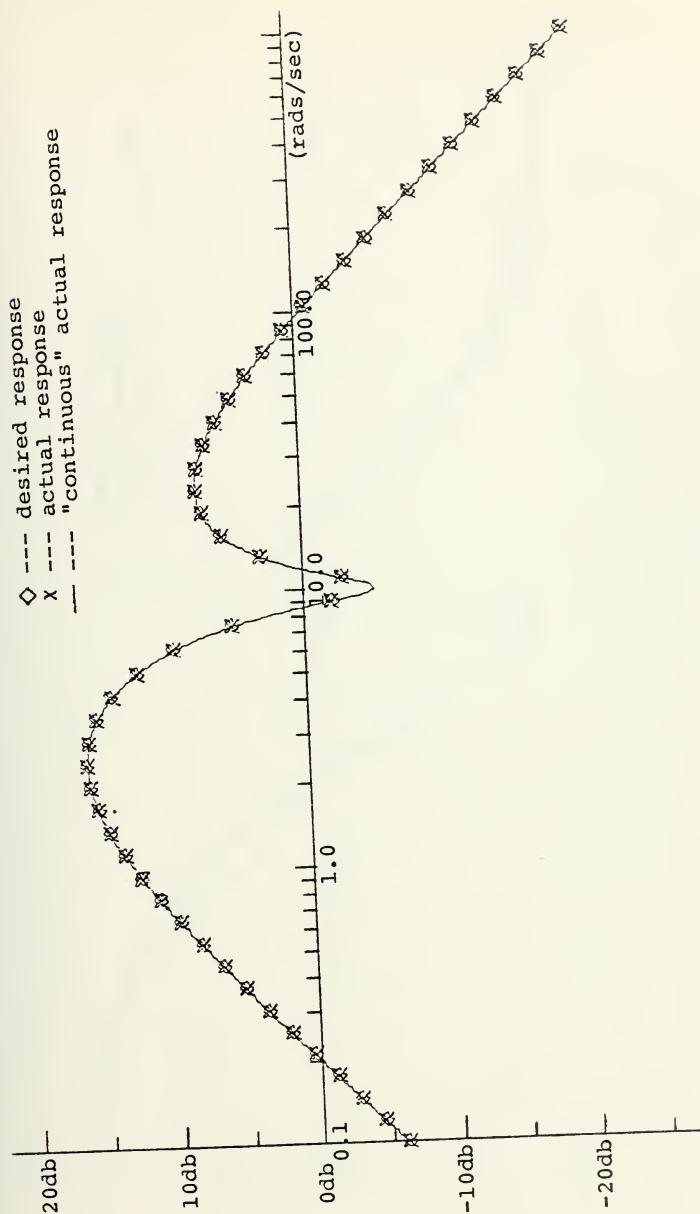
Figure V-3H

### 3. Synthesis Example 3.

A frequency response of a system with resonant peaks, shown in figure V-4, is considered in this example. The same example is considered in ref. 19, where a combination of graphical and analytical techniques are used to determine the transfer function. Initially a transfer function consisting of a first order numerator and second order denominator was assumed in trying to model the system. This choice was based on the low and high frequency magnitude curve slopes of approximately +20 db/decade and -20 db/decade respectively and the low and high frequency phase values of approximately +90 degrees and -90 degrees. This form failed to match the dip in the magnitude curve in the region of 10 radians/second and the jump in the phase curve in the same frequency area. After increasing the order of the numerator and denominator, a transfer function consisting of a third order numerator and fourth order denominator was found to provide a phase and gain curve that closely approximates the measured frequency response. The results of this are shown in figures V-4A through V-4H. The numerical values of the transfer function parameters returned from the program closely approximate those of the actual system from which the frequency response measurements were obtained. The interested reader may verify this fact by comparing the values shown in figures V-4G and V-4H with those of the actual system given in ref. 19.

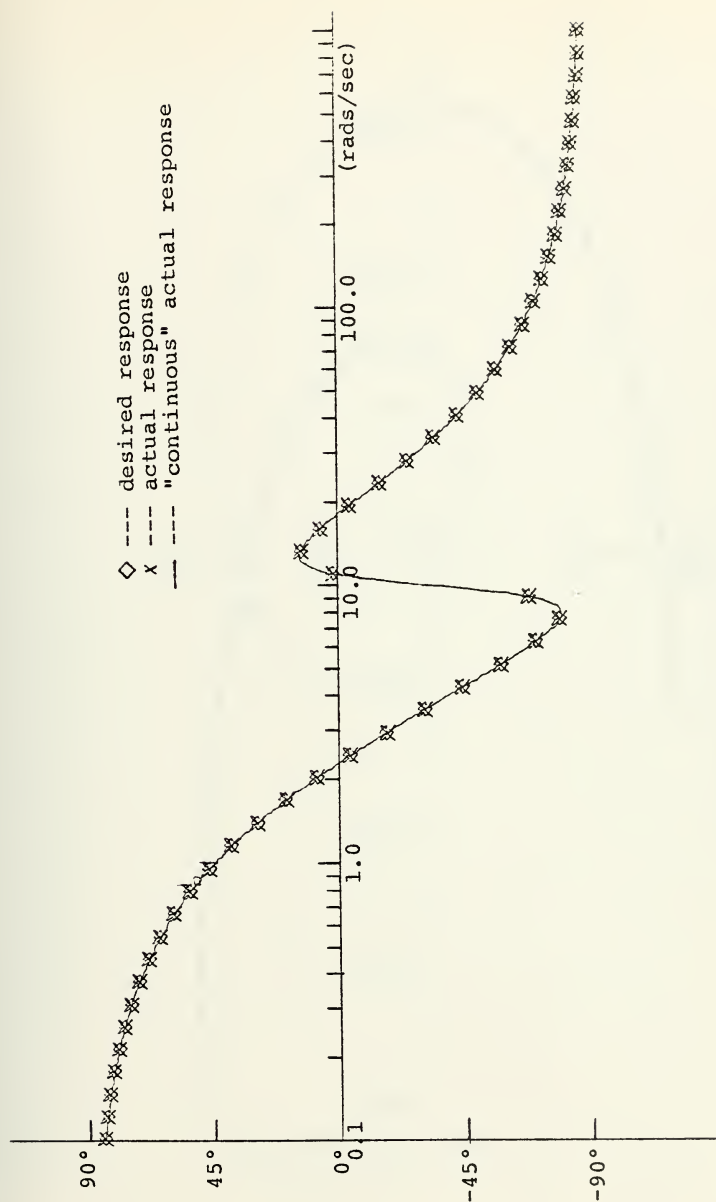


Input Frequency Response Data  
Figure V-4



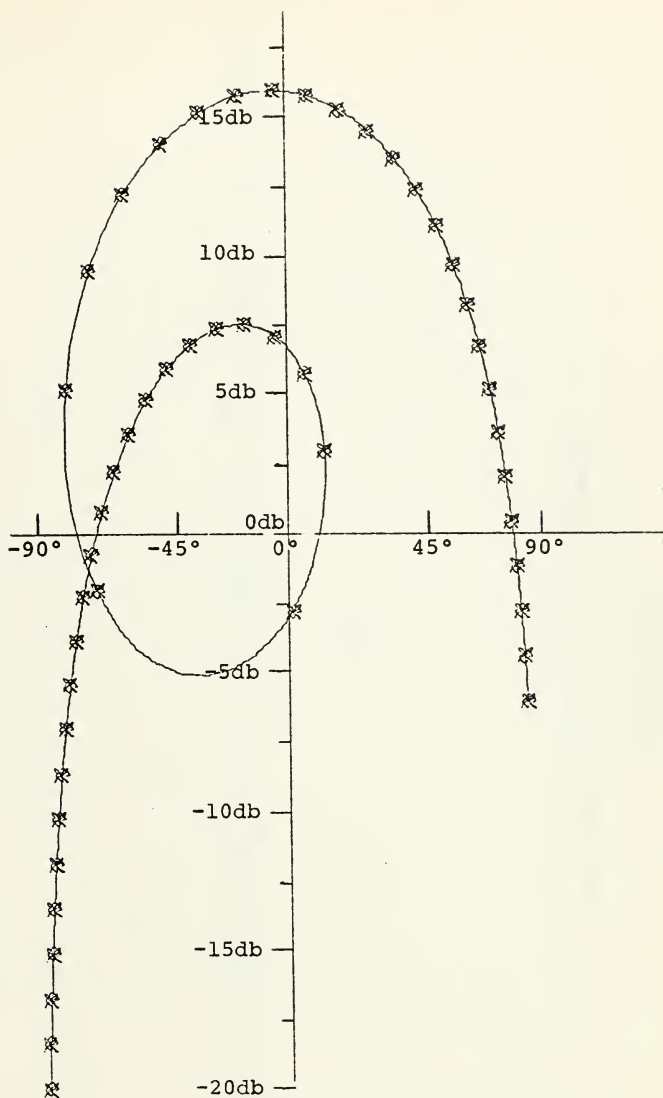
Magnitude vs. Frequency  
Figure V-4A





Phase vs. Frequency

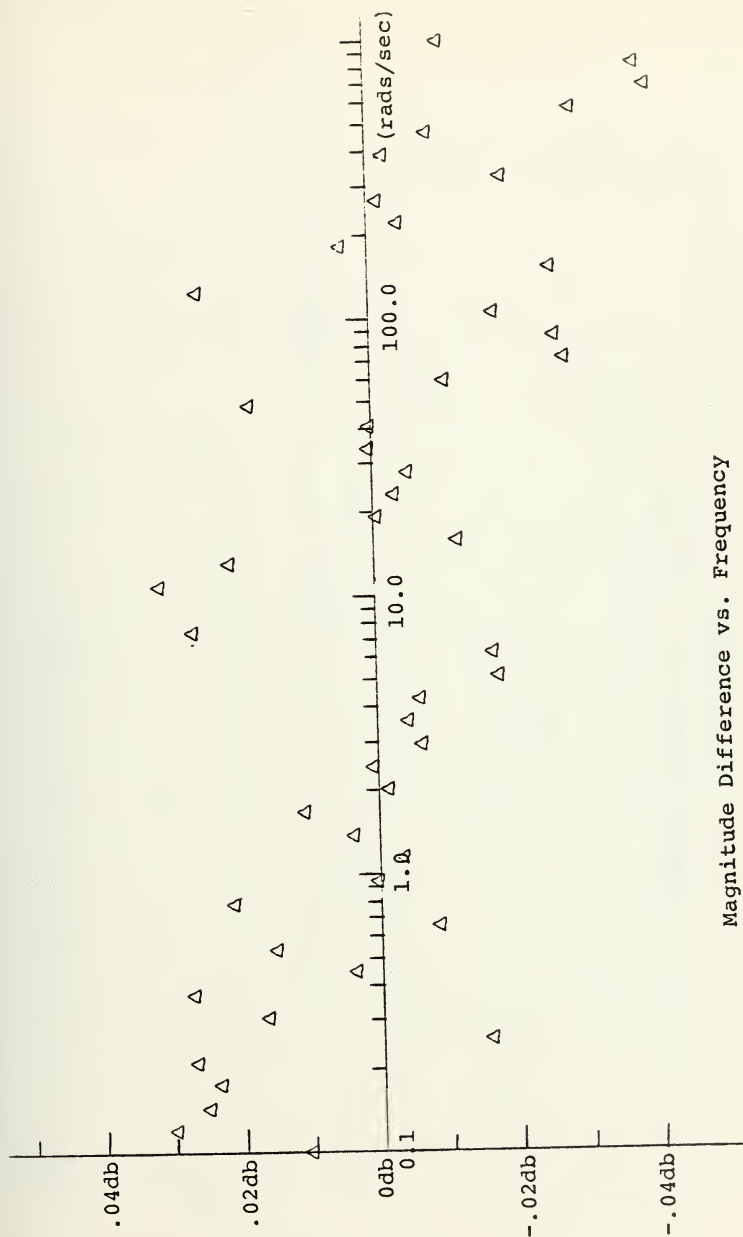
Figure V-4B

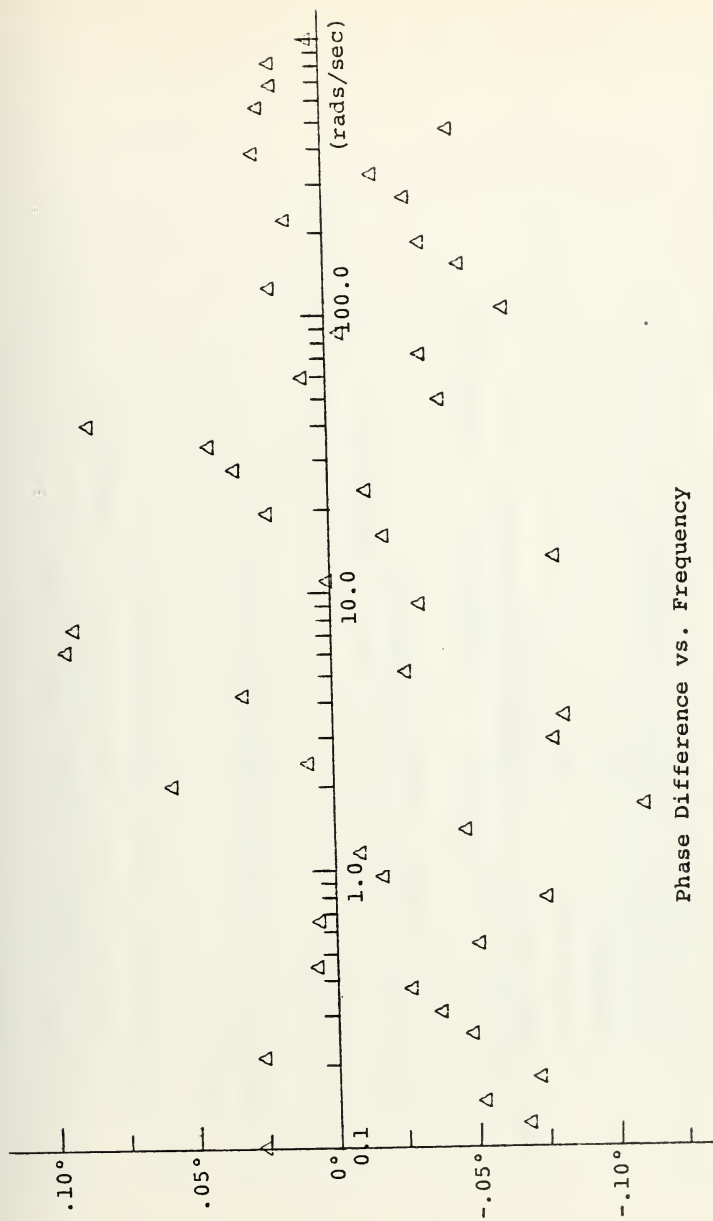


Magnitude vs. Phase

Figure V-4C







Phase Difference vs. Frequency

Figure V-4E

TITLE --- SYNTHESIS EXAMPLE THREE 50 PTS

UNCOMPENSATED TRANSFER FUNCTION GAIN = 1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

COMPENSATOR TRANSFER FUNCTION GAIN = 1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 3.000000E 00 3.000000E 00 1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR-ROOTS  
ARE: REAL PART IMAGINARY PART

-1.000000E 00 0.0

-1.000000E 00 0.0

-1.000000E 00 0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 4.000000E 00 6.000000E 00 4.000000E 00 1.000000E 00

Computer Numerical Output

Figure V-4F

COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS  
ARE: REAL PART IMAGINARY PART

-1.000000E 00 0.0  
-1.000000E 00 0.0  
-1.000000E 00 0.0  
-1.000000E 00 0.0

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE  
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

THE TOTAL NUMBER OF TRIALS CALLED FUP = 10000

THE CCST FUNCTION TO BE USED IS THE TYPE 0

THE MINIMUM CCST FUNCTION VALUE = 4.754462E-04

THE ERROR RETURN CODE FROM BOXPLX = -1

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

6.788290E 00 2.365316E 05 4.754176E 03 2.362180E 03

Computer Numerical Output

Figure V-4G

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
ROOTS ARE: REAL PART IMAGINARY PART

-1.006292E 00 5.955896E 00  
-1.006292E 00 -9.955896E 00  
-2.869925E-05 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

4.726638E 04 4.021491E 04 9.944633E 03 8.747952E 02 2.364240E 01

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
ROOTS ARE: REAL PART IMAGINARY PART

-1.991656E 01 0.0  
-1.012778E 01 0.0  
-4.556751E 00 0.0  
-1.999504E 00 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 9.991237E 01

THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES  
THAT THE SYSTEM IS STABLE

Computer Numerical Output  
Figure V-4H

#### 4. Synthesis Example 4.

In this final example problem a low order approximation to a known higher order system is illustrated. The original system transfer function under consideration consists of a seventh order numerator and an eight order denominator given by

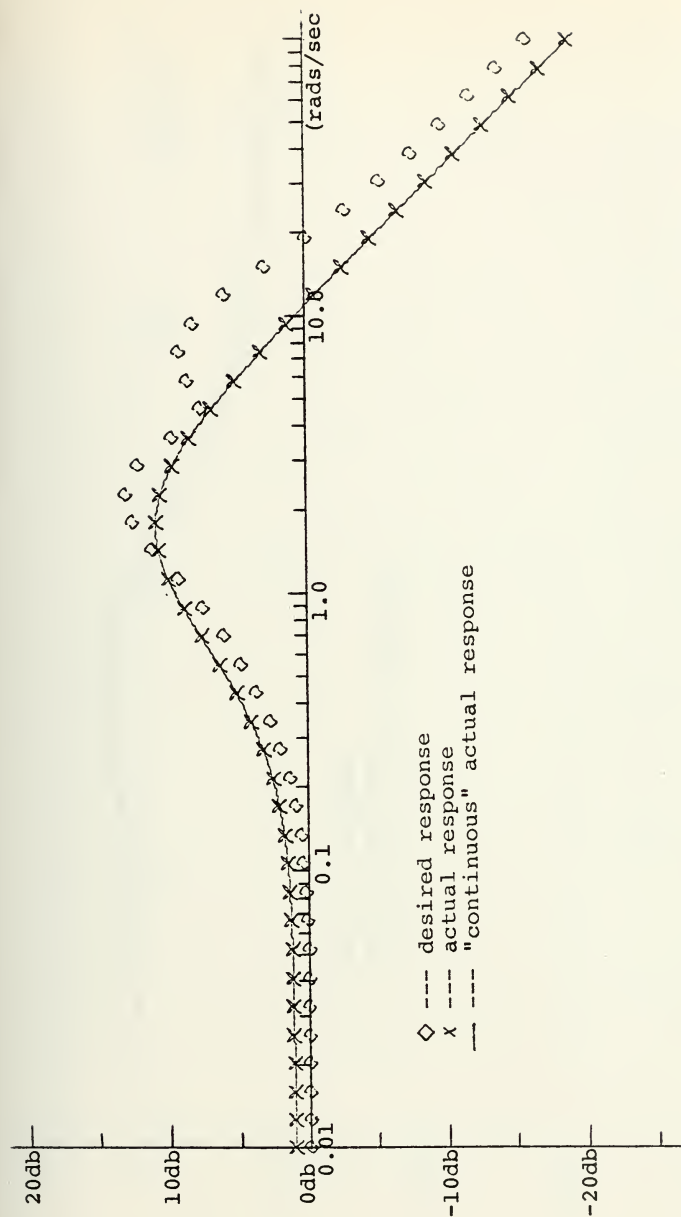
$$16s^7 + 483s^6 + 6010s^5 + 36380s^4 + 122664s^3 + 222088s^2 + 185760s + 40320$$

---


$$s^8 + 36s^7 + 546s^6 + 4536s^5 + 22449s^4 + 67284s^3 + 118124s^2 + 109584s + 40320$$

An analytical method of approximating this high order transfer function is discussed in ref. 21. Initially 40 discrete points, representing the frequency response of the high order transfer function were selected over the frequency range from 0.01 radians/second to 100.0 radians/second. In order to be able to form a comparison with the reduced order model obtained in ref. 21, a transfer function consisting of a first order numerator and a second order denominator was assumed. The results returned from the program using this form of transfer function are shown in figures V-5A through V-5H. It is not surprising that there is a significant amount of deviation in the magnitude curve in the vicinity of the resonant effects of the higher order system nor that the phase response of this reduced order model does not match the higher order phase response in the higher frequency range. In order to compare these results with analytical methods for forming reduced order models, the frequency response for the reduced order model of ref. 21 is shown in figures V-5I through V-5M. The numerical values returned from the CALICO program are of the same order of magnitude as those found by the analytical technique, but as can be seen from the magnitude curves of

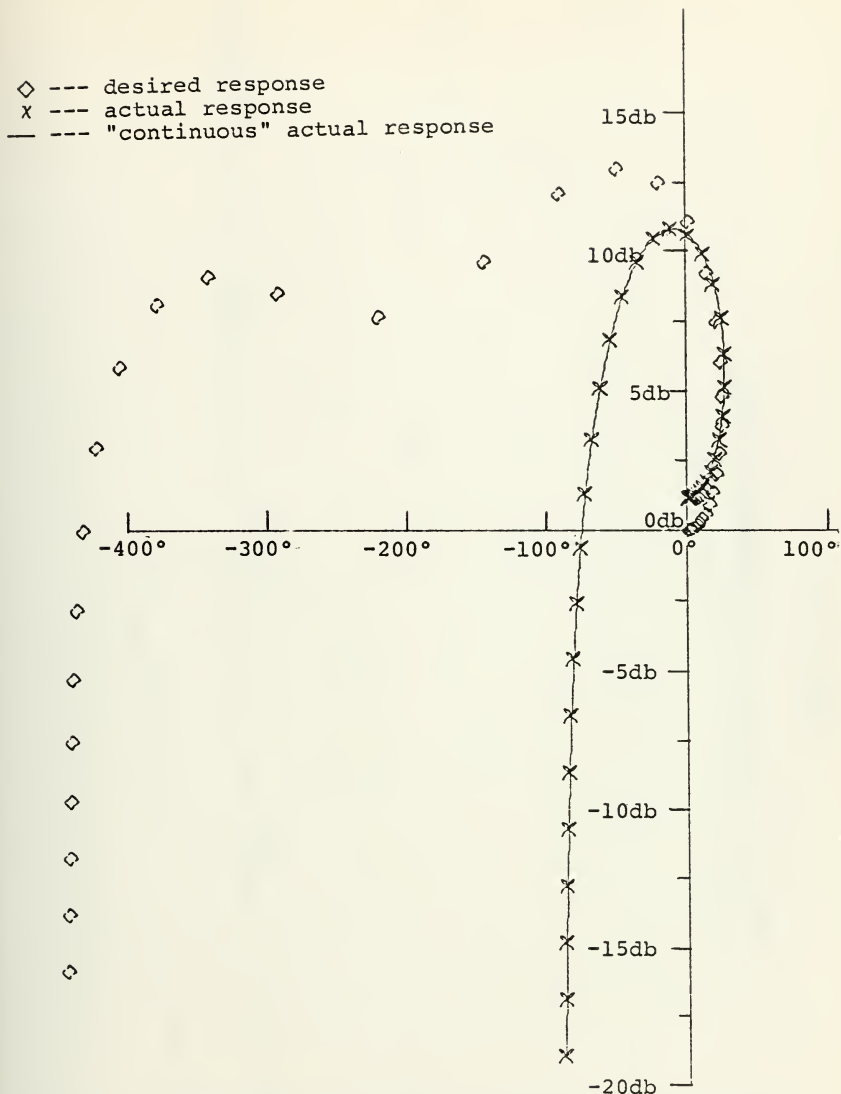
the two approaches (figures V-5A and V-5I), the analytical method matches the higher order transfer function extremely well at the lower and higher frequencies and makes little attempt to match the higher order system in the range of frequencies at which the resonant peaks occur. The results returned from the program, on the other hand because they are a function of the deviation of the response from the desired values over the entire range of frequencies, tend to distribute any error between the responses over the entire range of frequencies being considered. A slightly different representation of the system may be obtained by varying the frequency range under consideration and the type of cost function used in the program. For example, figures V-5N through V-5U illustrate the effects of using a type two cost function and limiting the frequency range from 0.1 to 100.0 radians/second. Again the decision as to which model is best must be made by the user in light of the intended use of the reduced order representation.



Magnitude vs. Frequency, Run #1  
Figure V-5A

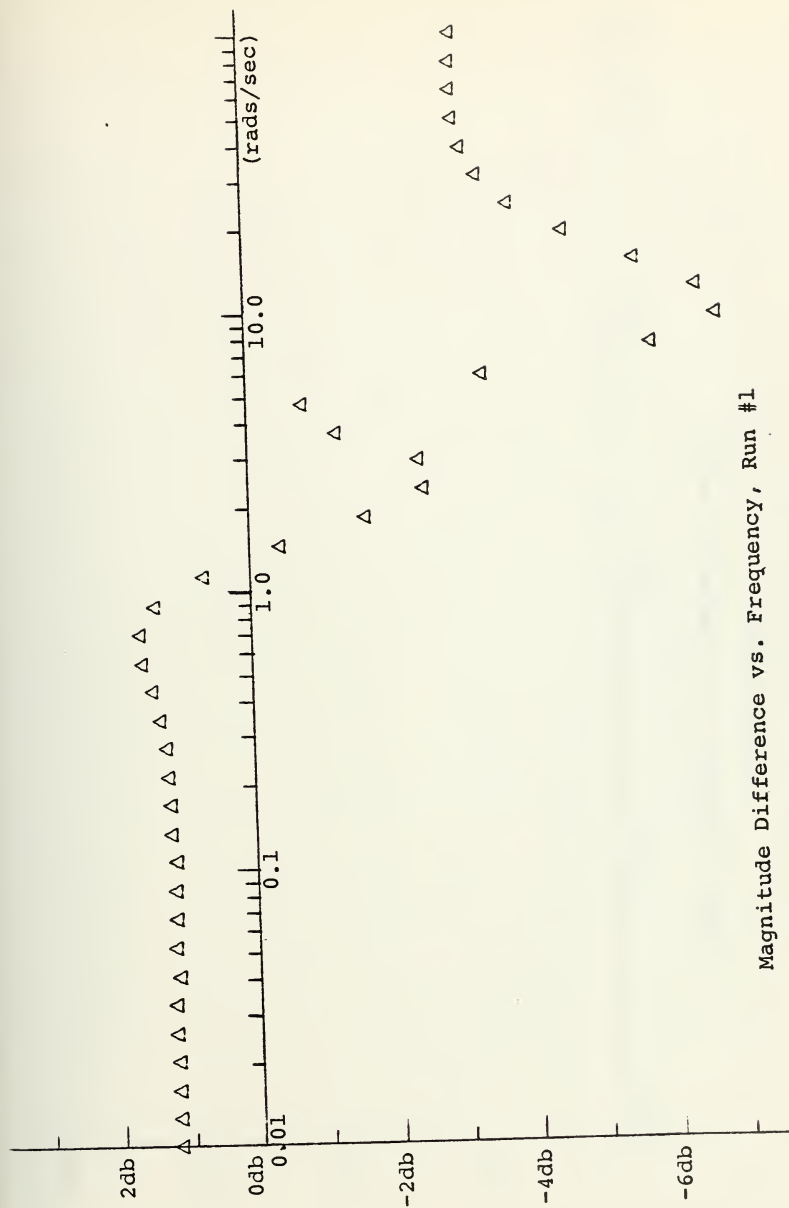






Magnitude vs. Phase, Run #1

Figure V-5C



Magnitude Difference vs. Frequency, Run #1

Figure V-5D

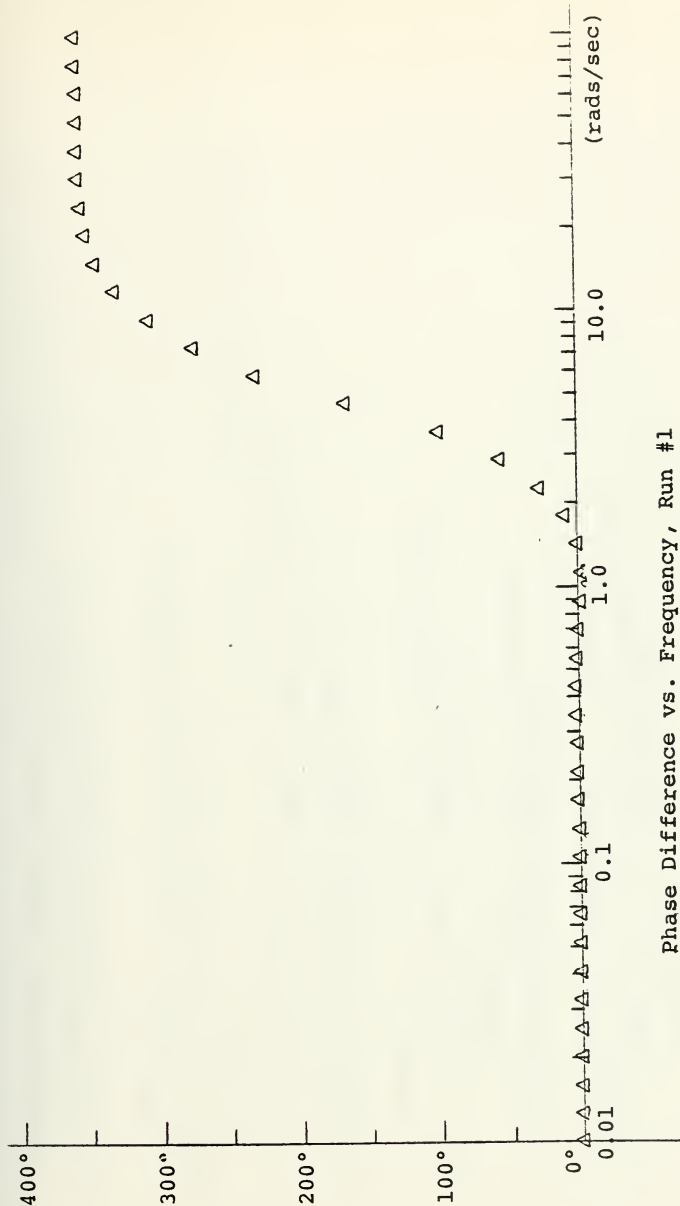


Figure V-5E

-----  
TITLE --- LCM CRC APPROX EXAMP 40PTS  
-----

UNCOMPENSATED TRANSFER FUNCTION GAIN = 1.000000E 00  
UNCOMPENSATED TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00  
-----

COMPENSATOR TRANSFER FUNCTION GAIN = 1.000000E 00  
COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS  
ARE: REAL PART IMAGINARY PART

-1.000000E 00 0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 2.000000E 00 1.000000E 00

COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS  
ARE: REAL PART IMAGINARY PART

-1.000000E 00 0.0

-1.000000E 00 0.0  
-----

Computer Numerical Output, Run #1  
Figure V-5F

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE  
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

THE COST FUNCTION TO BE USED IS THE TYPE 1

THE MINIMUM COST FUNCTION VALUE = 1.321693E 01

THE ERROR RETURN CODE FROM BOXFLX = 0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

2.14059E 05 -- 6.417439E 05

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
REAL PART IMAGINARY PART

-3.348156E-01 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.876209E 05 1.869405E 05 5.706169E 04

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
REAL PART IMAGINARY PART

-1.638056E 00 7.776945E-01

-1.638056E 00 -7.776945E-01

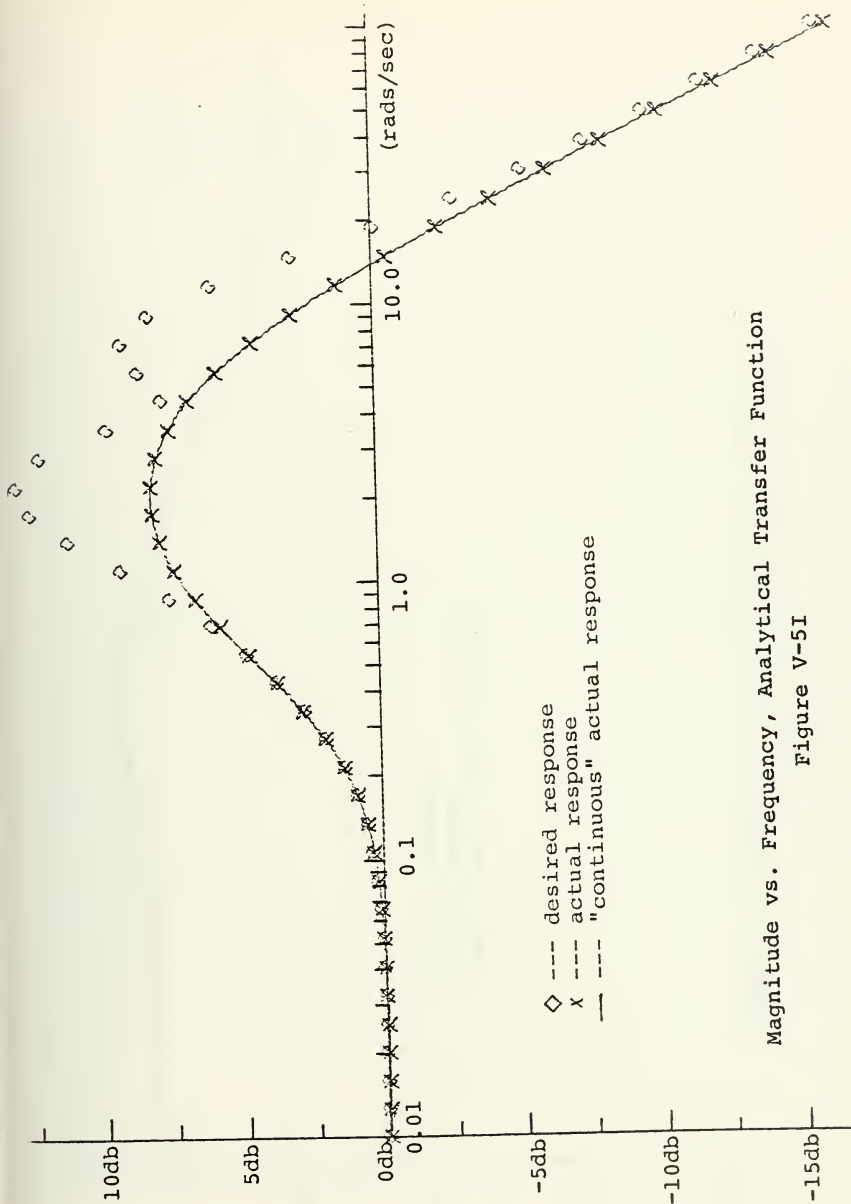
Computer Numerical Output, Run #1

Figure V-5G

FREQUENCY	MAGNITUDE(CB)	DESIED MAG(DB)	PHASE (DEG)	DESIED PHASE
9.555958E-03	1.181425E	0.0	1.398748	0.0
1.270000E-02	1.183658E	0.0	1.447248	0.0
1.600000E-02	1.187197E	0.0	1.822537	0.0
2.000000E-02	1.192967E	0.0	3.107444	0.0
2.570000E-02	1.202132E	0.0	3.922201	0.0
3.260000E-02	1.216521E	0.0	3.700172	0.0
4.120000E-02	1.249154E	0.0	4.663218	0.0
5.220000E-02	1.317746E	0.0	5.891981	0.0
6.609954E-02	1.326473E	0.0	7.390000E	0.0
8.379953E-02	1.425586E	0.0	9.268644	0.0
1.059999E-01	1.573788E	0.0	1.151783	0.0
1.339999E-01	1.792952E	0.0	1.416620	0.0
1.700000E-01	1.125519E	0.0	1.722145	0.0
2.150000E-01	2.566750E	0.0	2.044739	0.0
2.724999E-01	3.266556E	0.0	3.64001	0.0
3.460000E-01	4.121212E	0.0	3.255356	0.0
4.380000E-01	5.186727E	0.0	2.773955	0.0
5.540000E-01	6.385768E	0.0	2.751895	0.0
7.020000E-01	7.663657E	0.0	2.505437	0.0
8.890000E-01	8.853053E	0.0	1.997882	0.0
1.128999E-00	9.554955E	0.0	1.200836	0.0
1.428599E-00	1.067664E	0.0	1.683446	0.0
1.759999E-00	1.636059E	0.0	-1.007033	0.0
2.290000E-00	1.021539E	0.0	-2.253156	0.0
2.839999E-00	9.678394E	0.0	-3.749161	0.0
3.669999E-00	8.401459E	0.0	-4.546928	0.0
4.639999E-00	6.861783E	0.0	-5.321912	0.0
5.879999E-00	5.118952E	0.0	-6.163765	0.0
7.440000E-00	3.268562E	0.0	-6.749031	0.0
9.429999E-00	1.321501E	0.0	-7.219646	0.0
1.190000E-01	-6.150827E	0.0	-7.587126	0.0
1.510000E-01	-2.626412E	0.0	-7.895545	0.0
1.909999E-01	-4.648530E	0.0	-8.118462	0.0
2.420000E-01	-6.655740E	0.0	-8.304007	0.0
3.070000E-01	-8.741075E	0.0	-8.451254	0.0
3.889999E-01	-1.079027E	0.0	-8.566870	0.0
4.920000E-01	-1.282622E	0.0	-8.657525	0.0
6.299999E-01	-1.488786E	0.0	-8.729959	0.0
7.990000E-01	-1.695350E	0.0	-8.786683	0.0
1.000000E-02	-1.1895814E	0.0	-8.831483	0.0

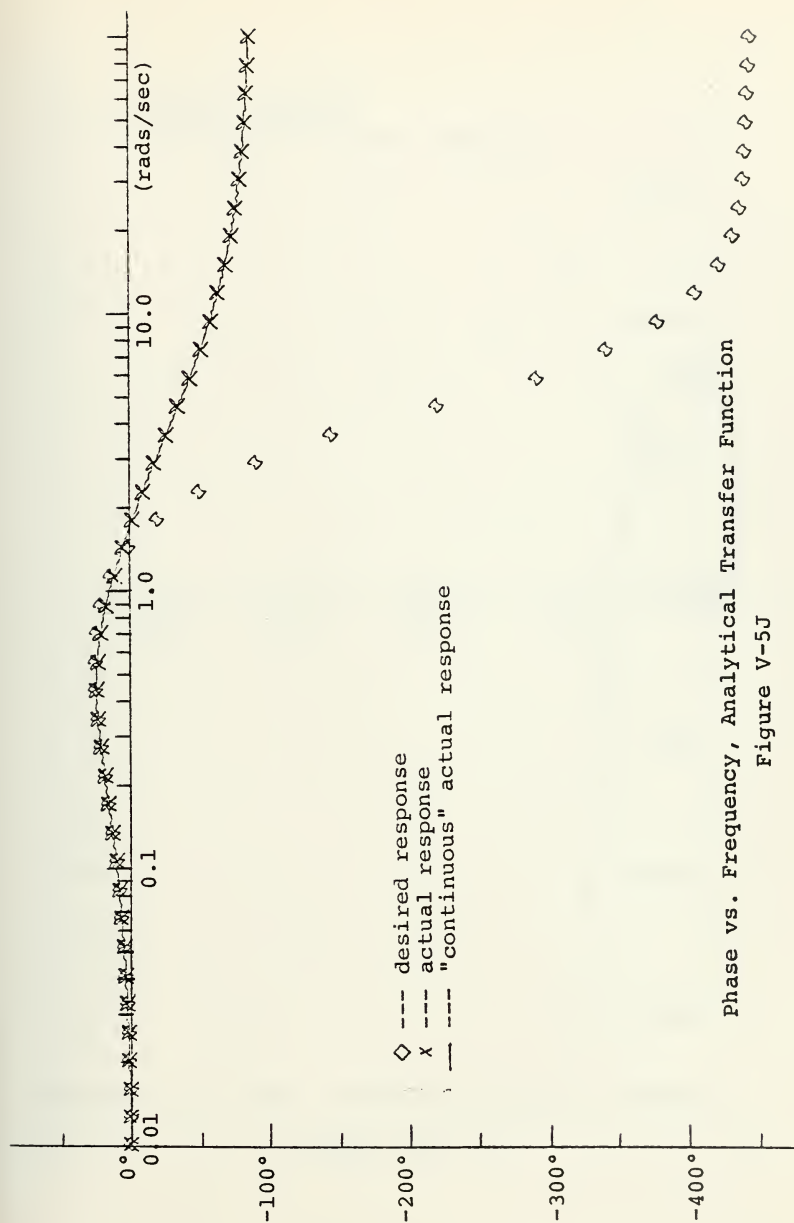
THE ROOT TEST OF THE CHARACTERISTIC EQUATION INDICATES  
THAT THE SYSTEM IS STABLE

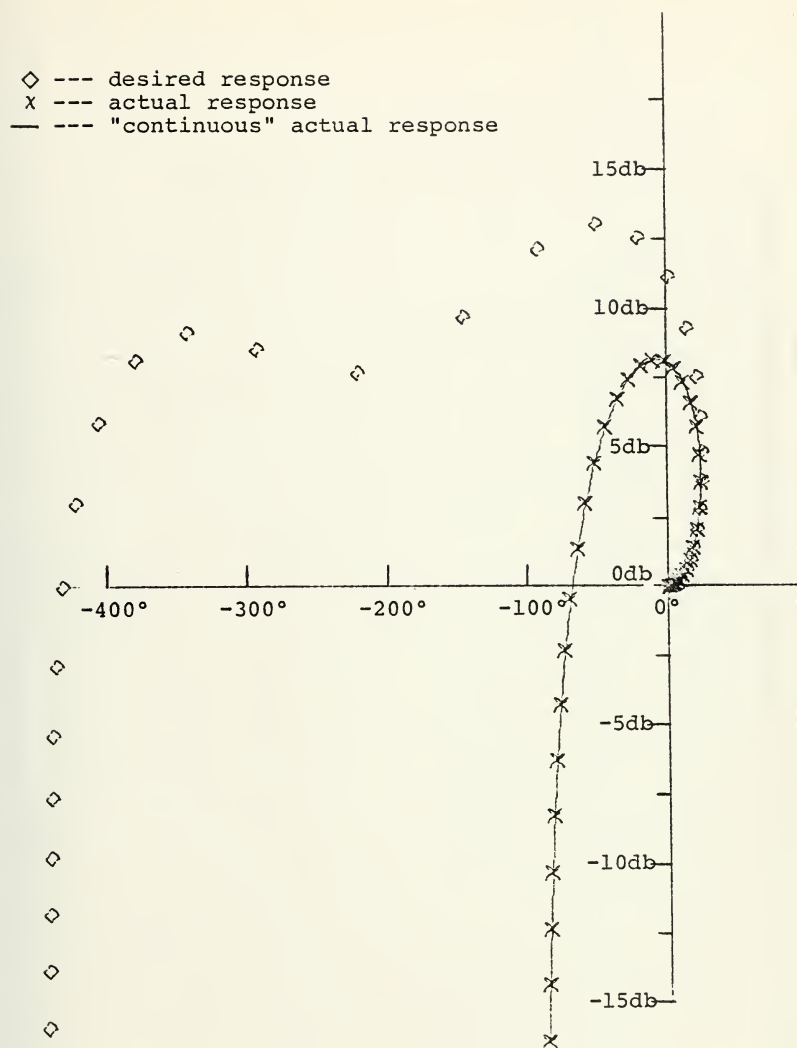
Computer Numerical Output, Run #1  
Figure V-5H



Magnitude vs. Frequency, Analytical Transfer Function  
Figure V-5I

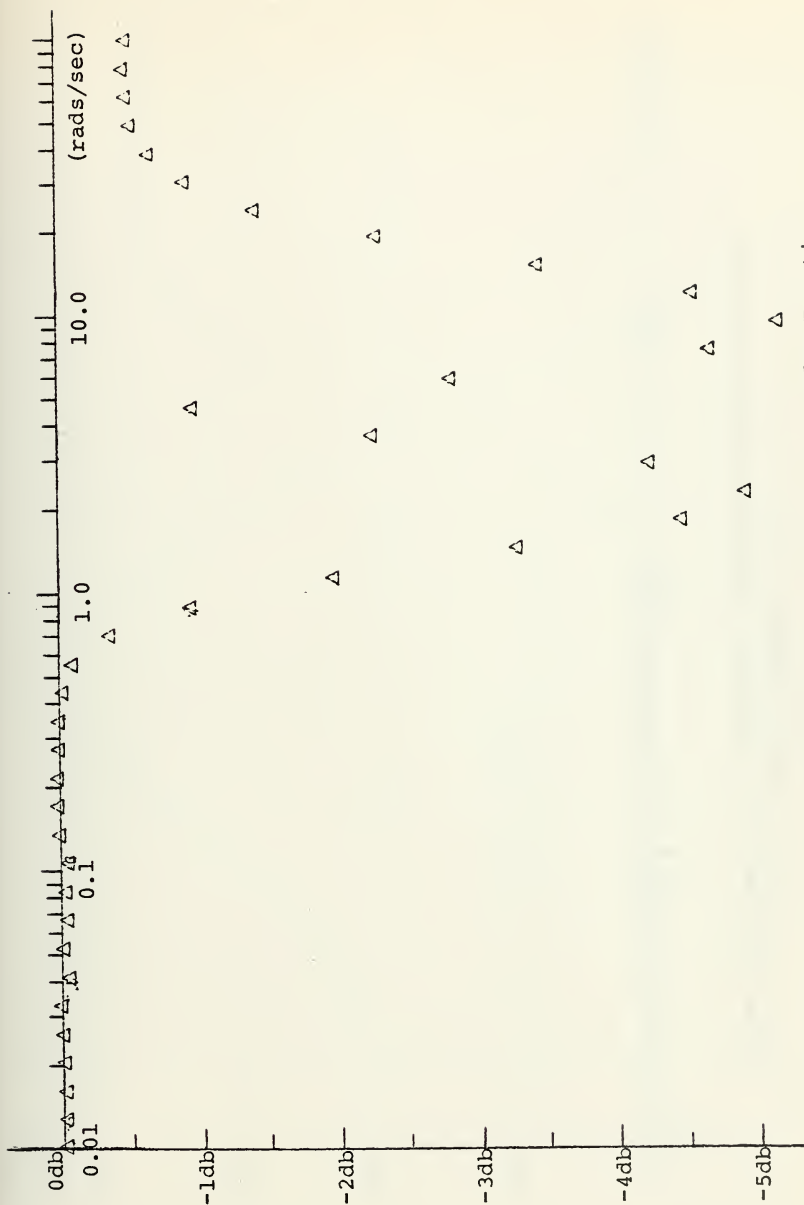






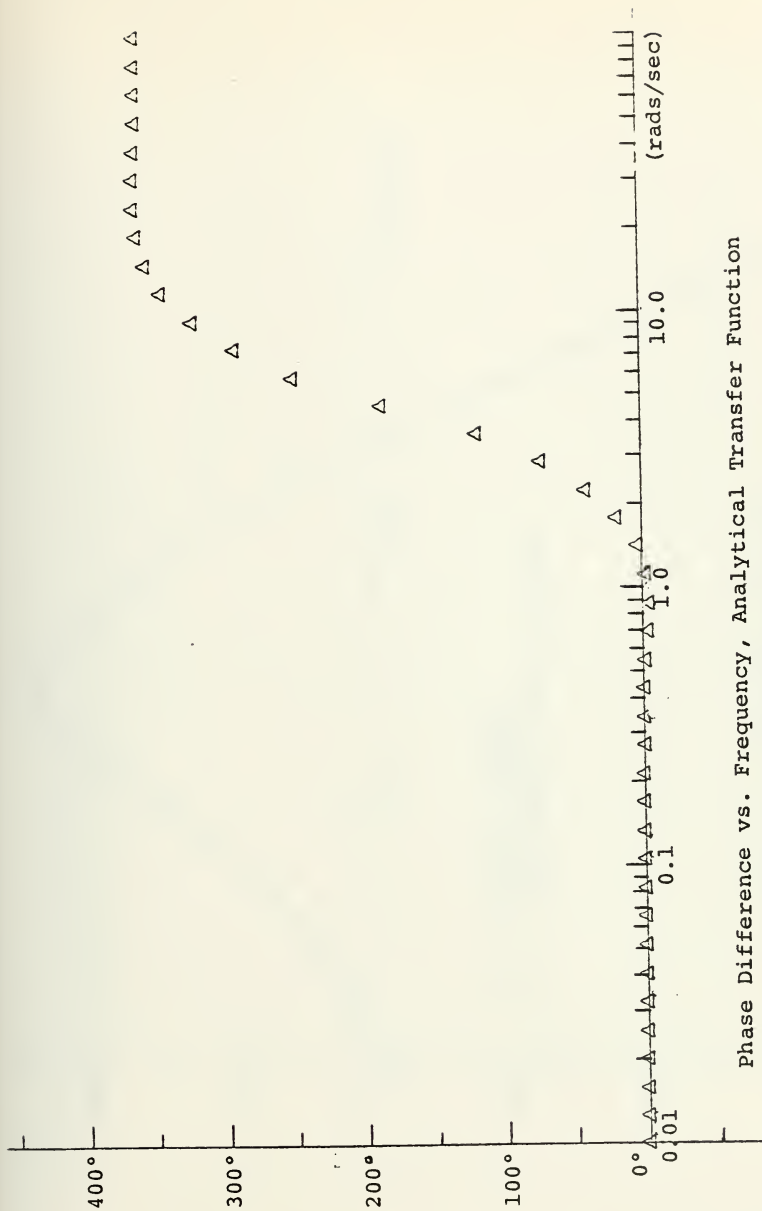
Magnitude vs. Phase, Analytical Transfer Function

Figure V-5K

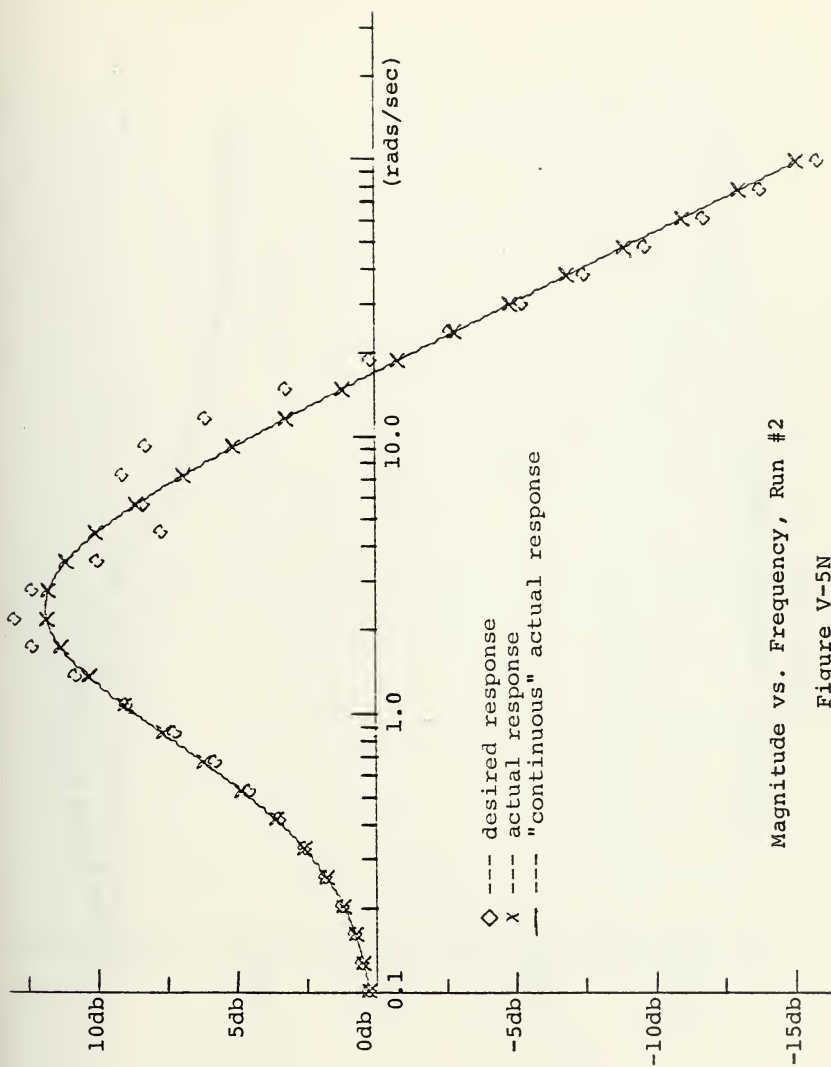


Magnitude Difference vs. Frequency, Analytical Transfer Function

Figure V-5L

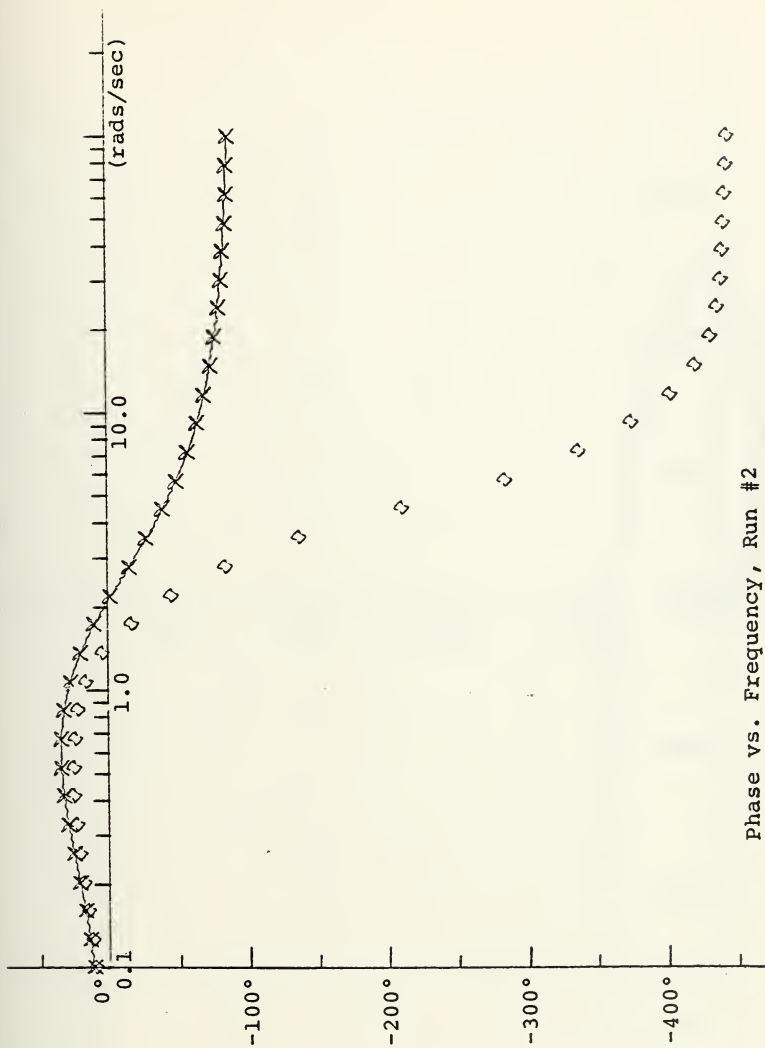


Phase Difference vs. Frequency, Analytical Transfer Function  
Figure V-5M



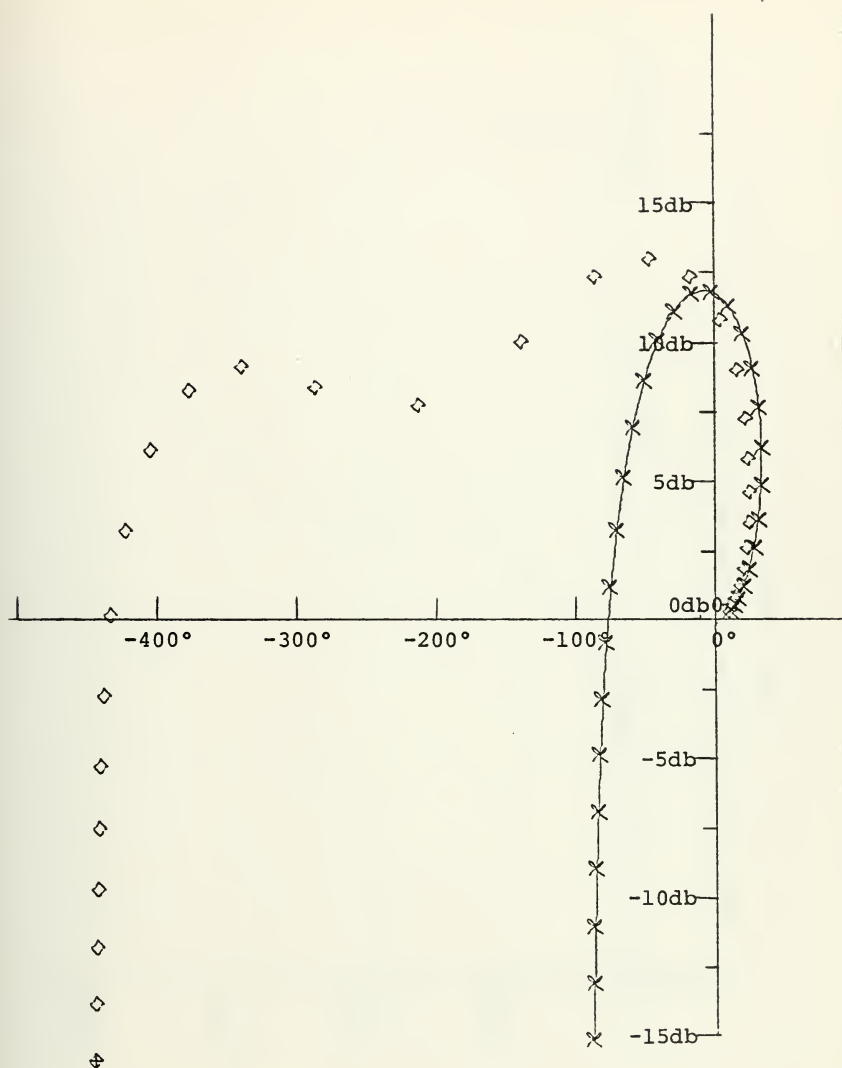
Magnitude vs. Frequency, Run #2

Figure V-5N



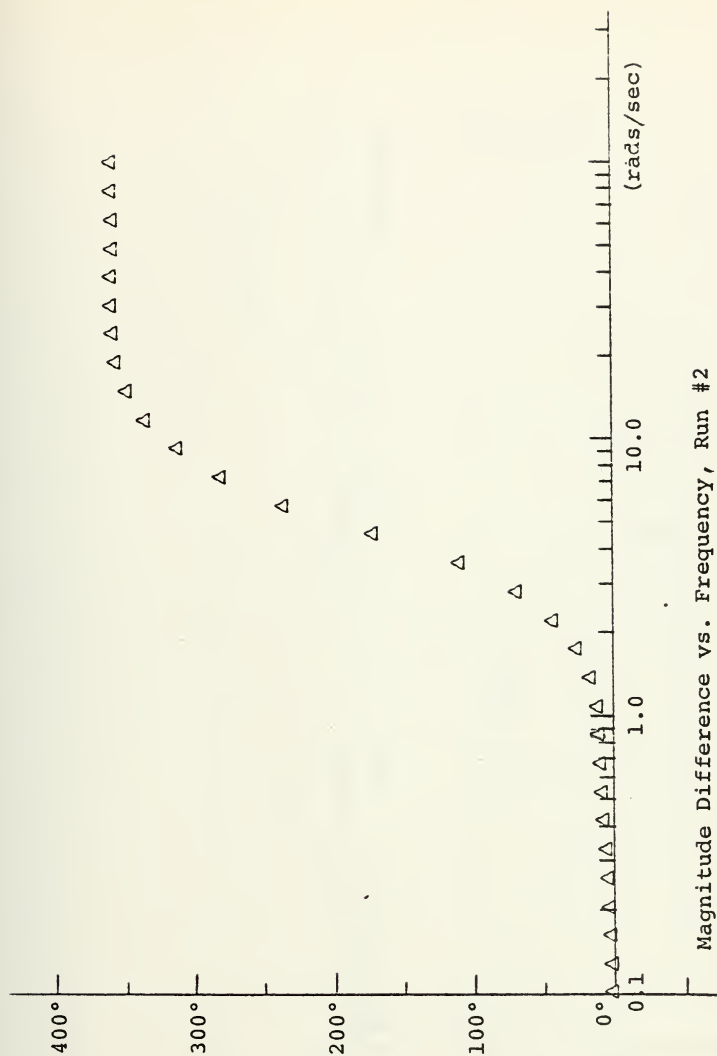
Phase vs. Frequency, Run #2

Figure V-50



Magnitude vs. Phase, Run #2

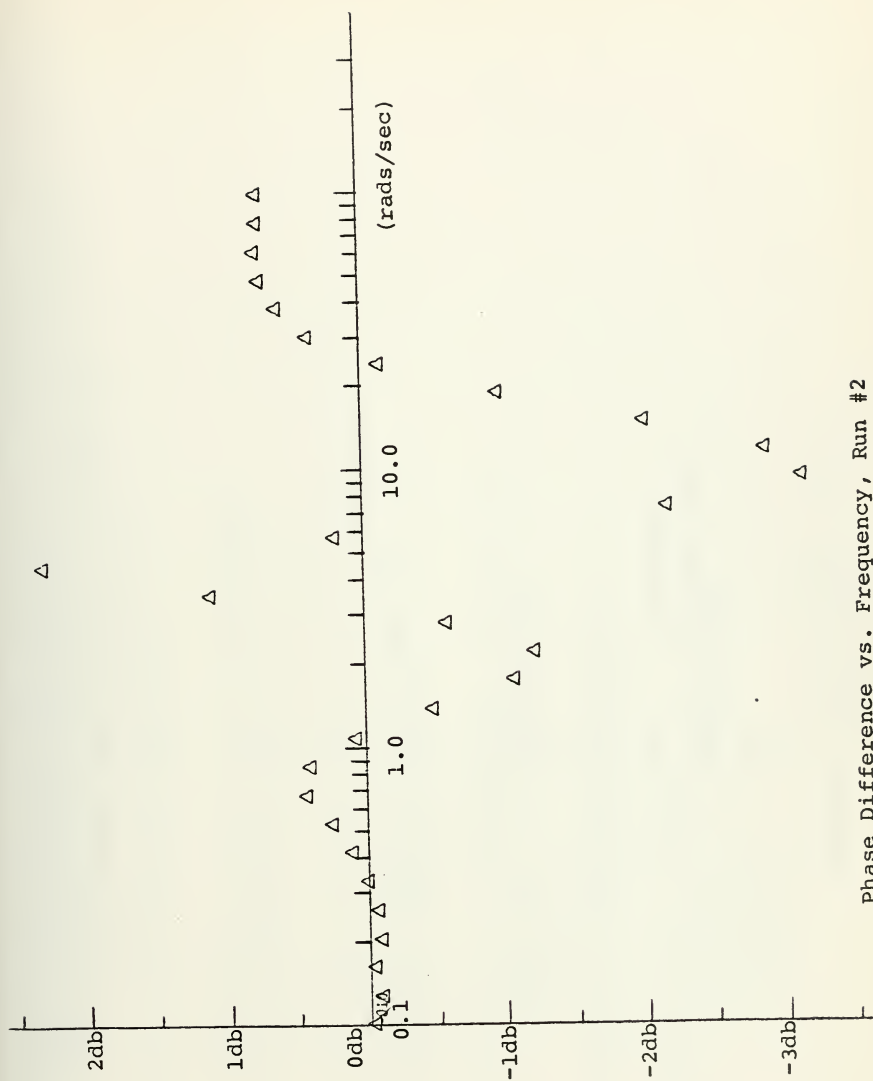
Figure V-5P



Magnitude Difference vs. Frequency, Run #2

Figure V-5Q





Phase Difference vs. Frequency, Run #2

Figure V-5R

TITLE --- LOW FRE APPROX EXPAND 30PTS

UNCOMPENSATED TRANSFER FUNCTION GAIN = 1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

COMPENSATED TRANSFER FUNCTION GAIN = 1.000000E 00

COMPENSATED TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 1.000000E 00

COMPENSATED TRANSFER FUNCTION NUMERATOR ROOTS  
ARE: REAL PART ..... IMAGINARY PART

-1.000000E 00 0.0

COMPENSATED TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00 2.000000E 00 1.000000E 00

COMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS  
ARE: REAL PART ..... IMAGINARY PART

-1.000000E 00 0.0

-1.000000E 00 0.0

Computer Numerical Output, Run #2

Figure V-5S

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE  
TYPE, THEREFORE AN RIGHT HALF PLANE ZERO'S WILL BE ALLOWED IN  
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

THE TOTAL NUMBER OF TOTALS CALLED FOR = 10000

THE COST FUNCTION TO BE USED IS THE TYPE 2

THE MINIMUM COST FUNCTION VALUE = 4.723737E-01

THE ERROR RETURN CODE FROM EXECLX = 2

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

3.427256E 05 9.702700E 05

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR  
FACTS ARE:  
REAL PART  
IMAGINARY PART

-3.532414E-01 0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
COEFFICIENTS IN ASCENDING POWERS OF S

3.429754E 05 2.500541E 05 5.576800E 04

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR  
FACTS ARE:  
REAL PART  
IMAGINARY PART

-2.241913E 00 1.060160E 00

-2.241913E 00 -1.060160E 00

Computer Numerical Output, Run #2

Figure V-5T



## VI. CONCLUSION AND SUGGESTED FUTURE STUDIES

The investigations carried out in this thesis illustrate the feasibility of automating the classical design of compensators by specifying a set of desired values for the magnitude and phase response of the open loop system at a number of discrete frequency values over the range of interest. The results achieved show that the complex minimization method of M. J. Box is quite capable of handling the constrained minimization problem and conveniently provides a means of avoiding the numerical differentiation of the cost function, required with gradient search techniques. The incorporation of stability criterion in the implicit constraints of the minimization algorithm also insures that the program will not design an unstable compensator. By specifying the desired response in terms of the magnitude and phase profiles, common frequency domain specifications such as open loop bandwidth, phase margin, and gain margin have been incorporated into one common format and the necessity for specialized routines to check these specifications have been avoided. Those example problems presented for which known results had been established by other techniques show that the algorithm presented provides accuracy that is certainly within the necessary tolerances for most engineering applications of compensator design.

Also, while not specifically designed for the purpose, the basic premise of the algorithm's minimizing the difference between some desired frequency response and the frequency response of the system as calculated by the computer program while the transfer function parameters are

varied has been shown to be effective in both determining a transfer function from measured frequency response data and in forming low order models of known higher order linear systems. In the formulation of low order models the choice of cost function and the frequency range specified may be used to emphasize different aspects of the frequency response which the low order model is to approximate.

While this work has shown that the basic algorithm presented can be used to effectively design series compensators based on frequency response specifications, considerable work remains to be done in this area. Specifically, an option to select that only real poles and zeros be considered as viable solutions to the compensator design algorithm would not be a difficult addition to the program and should prove quite useful. Also, the incorporation of a provision for handling pure time delays would not require significant modification of the program coding and would extend the capability of the algorithm to the solution of problems involving transport delays. Investigation of the method presented should be further pursued in terms of multiloop systems and systems involving compensation in the feedback loop. It may also prove worthwhile to investigate extending this method to nonlinear components using describing function techniques.

In the areas of transfer function synthesis and reduced order modeling, a more detailed investigation of both the effects of different cost functions and considering different frequency ranges should be conducted. In addition, it appears feasible to weight the cost function over a particular frequency range by varying the density of the discrete frequencies at which the desired magnitude and phase profiles are specified. Time did not permit a detailed investigation of this idea and further investigation into this area may be applicable to the

program utilization in both the areas of compensator design and transfer function synthesis.

## APPENDIX A

### DESCRIPTION OF SUBROUTINES

In this section a brief description and background of various subroutines is presented to aid the interested reader in better understanding the internal logic of the program. Also these descriptions are intended to facilitate the task of anyone desiring to make modifications or additions to the program in order to handle systems of a configuration other than the unity feedback form assumed throughout this work.



# SUBROUTINE CONORM

The purpose of this subroutine sub-program is to normalize the coefficient of the highest order term to unity. This is accomplished in a simple, straight forward manner by dividing all the coefficients of the polynomial by the coefficient of the highest order term. Thus a polynomial given by

$$P(s) = \sum_{i=0}^n a_i \cdot s^i$$

becomes on return from this subroutine

$$P(s) = \sum_{i=0}^n \frac{a_i}{a_n} \cdot s^i$$

where  $a_n$  is the coefficient of the  $s^n$  term. The normalizing factor  $a_n$  is also returned to the calling program.

It should be noted that the coefficients of the normalized polynomial are returned under the same variable name as the input coefficients.

Definitions of the input and output parameters are:

## Input Variables

V ----- A real one-dimensional array containing the coefficients of the polynomial whose highest order coefficient is to be normalized to unity. Upon return from the subroutine this array will contain the normalized coefficients of the polynomial.

N ----- The dimension of the array containing the  
coefficients to be normalized.

VNOEM ----- The normalizing factor, returned to the  
calling program, by which all coefficients have been  
divided.

# SUBROUTINE PHASE

This subroutine sub-program is used to calculate the phase angle of a polynomial at a particular frequency. In order to circumvent the difficulty associated with standard arctangent routines, which return values limited to the range  $-\pi \leq \theta \leq \pi$ , the phase contributions from the individual factors are calculated and then summed to form the total phase angle at a particular frequency. That is, if the polynomial is factored and represented as the product of first order terms of the form

$$P(s) = \prod_{i=0}^n (s + z_i)$$

where in general  $z_i$  may be a complex number of the form  $z_i = a_i + jb_i$ , then at a particular frequency  $s = j\omega$  the phase contribution of the individual factors may be evaluated as

$$\phi_i = \tan^{-1} \frac{(b_i + \omega)}{a_i}$$

and the total phase angle at the particular frequency is the sum of these individual phase contributions. The phase contribution of each of these first order terms will be within the principal values of the arctangent routine of the computer and the problems generally associated with determination of the phase for higher than first order terms are avoided.

Definitions of the input and output parameters are:

#### Input Variables

RE ----- A real one-dimensional array containing the real parts of the roots of the polynomial.

RI ----- A real one-dimensional array containing the imaginary parts of the roots of the polynomial.

N ----- An integer value specifying the number of roots of the polynomial.

CMEGA ----- The frequency in radians/sec at which the value of the phase is to be computed.

#### Output Variables

TANGD ----- The total phase contribution of the polynomial given in degrees.

# SUBROUTINE PINVRT

The purpose of this subroutine sub-program is to rearrange the coefficients of a polynomial in inverse order from that fed into the subroutine. Consider for example a polynomial arranged in descending powers of s

$$F(s) = a_n s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0$$

Upon return from this subroutine the coefficients would be arranged in ascending powers of s. That is the polynomial would be arranged as follows

$$F(s) = a_0 + a_1 s + \dots + a_{n-1} s^{n-1} + a_n s^n$$

While both representations are mathematically equivalent the sequence of the coefficients must be taken into account when using various polynomial manipulation techniques within the computer. The method used in reversing the sequence of the input coefficients consists of temporarily storing the first coefficient and sequentially shifting the remaining coefficients. The coefficient in temporary storage is then placed in the last coefficient location. This process is continued with the use of nested do loops and each time the shifting operation occurs the coefficient in temporary storage is placed one location lower than the previous number retrieved from temporary storage.

This subroutine requires no other subroutines or functions to carry out the procedure. The size of the one dimensional array of coefficients that may be rearranged is determined by the dimension of the array in the calling program.

Definitions of the input and output parameters are:

## Input Variables

X(I) ----- A real one dimensional array containing the ccoefficients to be rearranged. Upon return from the subroutine this array contains the input coefficients in reverse sequence from that entered.

N ----- The dimension of the array containing the ccoefficients to be rearranged.

# SUBROUTINE PMPY

The purpose of this subroutine sub-program is to multiply the coefficients of one polynomial  $X(s)$  times the coefficients of another polynomial  $Y(s)$  to obtain the coefficients of the resulting polynomial  $Z(s)$ . That is, if it is desired to multiply two polynomials of the form

$$P_1(s) = a_0 + a_1s + a_2s^2 + \dots + a_{n-1}s^{n-1} + a_ns^n$$

and

$$P_2(s) = b_0 + b_1s + b_2s^2 + \dots + b_{m-1}s^{m-1} + b_ms^m$$

together in order to obtain a third resultant polynomial of the general form

$$P_3(s) = c_0 + c_1s + c_2s^2 + \dots + c_{i-1}s^{i-1} + c_is^i$$

this subroutine is used in accomplishing this. The order of the resulting polynomial is calculated by forming the sum of the orders of the two input polynomials as

$$i = n + m$$

The various coefficients of the resulting polynomial, which are returned in ascending powers of the independent variable, are calculated as the sums of products of the input coefficients by means of two nested do-loops. The arrangement of the two do-loops is such that the appropriate product terms are summed in forming the respective coefficients of the resultant polynomial. All input and output coefficients must be real numbers. The size of the polynomials that may be multiplied are controlled by their respective dimension statements within the calling program.

Definitions of the input and output parameters are:

### Input Variables

- X ----- A real one dimensional array containing the coefficients of the first polynomial to be multiplied. These must be arranged in ascending powers of the independent variable.
- IDIMX ----- The dimension of the array X containing the coefficients of the first polynomial.
- Y ----- A real one dimensional array containing the coefficients of the second polynomial to be multiplied. These must be arranged in ascending powers of the independent variable.
- IDIMY ----- The dimension of the array Y containing the coefficients of the second polynomial.

### Output Variables

- Z ----- A real one dimensional array containing the coefficients of the resultant polynomial. These coefficients are arranged in ascending powers of the independent variable.
- IDIMZ ----- The calculated dimension of the array Z containing the coefficients of the resultant polynomial.



# SUBROUTINE PVAL

The purpose of this subroutine sub-program is to evaluate a polynomial having the complex variable  $s = \sigma + j\omega$  and real coefficients  $a_i$ . For a given polynomial of order  $n$  of the form

$$P(s) = \sum_{i=0}^n a_i \cdot s^i$$

$s$  is declared as a complex variable within the subroutine and the indicated operations of summation and multiplication are carried out by a simple loop operation taking advantage of the associative and distributive laws of multiplication and addition. This subroutine requires no other sub-programs to perform its function.

Definitions of the input and output parameters are:

## Input Variables

A(I) ----- A real one-dimensional array containing the coefficients of the polynomial that is to be evaluated. The coefficients are assumed to be arranged in ascending order according to the powers of  $s$  with zero explicitly input for the coefficient of any missing powers of  $s$  occurring in the polynomial. The maximum size of this array is controlled by the dimension of the array within the main program. The values of the coefficients in this array remain unchanged after execution of the subroutine.

NN ----- The order of the polynomial to be evaluated.  
This value is one less than the dimension of A(I).

PR ----- The value of the real part of the complex  
variable s at which the polynomial is to be  
evaluated.

PI ----- The value of the imaginary part of the  
complex variable s at which the polynomial is to be  
evaluated.

#### Output Variables

VR ----- The real part of the complex resultant value  
of the polynomial evaluated at  $s = \sigma + j\omega$ .

VI ----- The imaginary part of the complex resultant  
value of the polynomial evaluated at  $s = \sigma + j\omega$ .

# SUBROUTINE ROUTH

The purpose of this subroutine sub-program is to determine if a given input polynomial possesses right half plane roots by using the Routh stability criterion. The polynomial coefficients are input in descending powers of the independent variable. From these coefficients the Routh array is calculated and the first column of this array is searched for an indication of the existence of right half plane roots. Thus from an input polynomial of the form

$$F(s) = b_n s^n + b_{n-1} s^{n-1} + \dots + b_1 s + b_0$$

the Routh array

1 <sup>st</sup> row	$b_n$	$b_{n-2}$	$b_{n-4}$	.....
2 <sup>nd</sup> row	$b_{n-1}$	$b_{n-3}$	$b_{n-5}$	.....
3 <sup>rd</sup> row	$a_{31}$	$a_{32}$	$a_{33}$	.....
4 <sup>th</sup> row	$a_{41}$	$a_{42}$	$a_{43}$	.....
	.	.	.	
	.	.	.	
	.	.	.	
	.	.	.	
	.	.	.	
n <sup>th</sup> row	$a_{n1}$	$a_{n2}$	$a_{n3}$	.....
(n+1) <sup>st</sup> row	$a_{(n+1)1}$	$a_{(n+1)2}$	$a_{(n+1)3}$	...

is formed, where in general the entries from the third row onward are given by

$$a_{ri} = \frac{[a_{(n-1)1}] [a_{(n-2)(i+1)}] - [a_{(n-2)1}] [a_{(n-1)(i+1)}]}{a_{(n-1)1}}$$

The first column of this array is then tested to determine if any of the elements are less than zero. Note that since all coefficients are limited to positive values by the search area restrictions in other parts of the program no provision has been made to handle the case of the coefficients of the polynomial being all negative. Thus a polynomial with all negative coefficients should be multiplied by -1.0 prior to using this particular subroutine for testing for right half plane roots.

Definitions of input and output parameters are:

#### Input Variables

Y ----- A real one dimensional array containing the coefficients of the polynomial in descending powers of the independent variable.

N ----- The dimension of the array Y.

#### Output Variables

ISTABL ----- Integer output indicating the absence or presence of right half plane roots

0---Indicates the polynomial has no right half plane roots.

1---Indicates the polynomial has right half plane roots.

# SUBROUTINE SEMEL

The purpose of this subroutine sub-program is to calculate the coefficients of a polynomial from the roots of the polynomial. The calculation is done using complex arithmetic in order to be able to compute the polynomial coefficients when complex roots are present. The resulting polynomial coefficients are, however assumed to be real. Thus, if complex roots are present they must be in conjugate pairs. Also the coefficient of the highest order term is set equal to unity. Consider a polynomial of degree n having n roots. This may be expressed as

$$P(s) = \prod_{i=0}^n (s + z_i)$$

where in general the  $z_i$ 's may be complex. The coefficients of the polynomial that results when the above indicated multiplication is performed may be computed by the formula

$$A_i = (-1)^{n-i} \cdot \sum (\text{product of roots taken } n-i \text{ at a time})$$

Thus the coefficients of the polynomial are given by

$$a_0 = z_1 \cdot z_2 \cdot z_3 \cdot z_4 \cdot \dots \cdot z_n$$

$$a_1 = z_1 \cdot z_2 \cdot z_3 \cdot \dots \cdot z_{n-1} + z_2 \cdot z_3 \cdot z_4 \cdot \dots \cdot z_n$$

.  
 .  
 .  
 .  
 .  
 .  
 .  
 .  
 .

$$a_{n-1} = z_1 + z_2 + z_3 + z_4 + z_5 + \dots + z_n$$

$$a_n = 1$$

The largest order polynomial that this particular subroutine will handle is a 20<sup>th</sup> order. If larger polynomials are to be considered the dimension of R within the subroutine must be changed.

Definitions of the input and output parameters are:

#### Input Variables

RR ----- A real one-dimensional array containing the real parts of the polynomial roots.

RI ----- A real one-dimensional array containing the imaginary parts of the polynomial roots.

N ----- An integer specifying the order of the polynomial or equivalently the dimension of the arrays containing the real and imaginary parts of the roots.

#### Output Variables

CF ----- A real one-dimensional array containing the

resulting ccoefficients of the polynomial arranged in ascending powers of the independent variable. The dimension of this array must be one larger than the dimensions of the arrays containing the real and imaginary parts of the roots.

APPENDIX B

PROGRAM LISTING





```

C VALUES OF THE FACTORS NEED BE ENTERED ONLY ONCE
C
3 IF (RI) 2,4,3
  I = I+1
  RTR(I) = -PR
  RTI(I) = -RI
4 IF (I) 1,N GO TO 2
  CALL SEMBL (N,RTR,RTI,CO)
5 CALL IC 8
6 READ (5,64) (CO(I),I=1,N)
  CC(NP) = 1.0
7 CALL PRCD (CO,NP,RTR,RTI,COF,NRCCT1,IERRT1)
  WRITE (6,67) NROOT1,IERRT1
  IF (IERRT1.EQ.0) GC TO 8
  CALL PRBM (CO,NP,RTR,RTI,COF,NRCCTA,IERRTA)
  WRITE (6,62)
  WRITE (6,63)
  WRITE (6,68) NRCOTA,IERRTA
  WRITE (6,63)
  WRITE (6,62)
8 GC TO (5,12,16,19), KCUNT
C MULTIPLY PLANT NUMERATOR COEFFICIENTS BY PLANT GAIN AND LOAD
C PLANT NUMERATOR ARRAY.
C
9 DO 10 I=1,NP
  A(I) = CC(I)*GAIN
10 CC CONTINUE
C
  NTFN = N
  NLE = NP
  WRITE (6,69)
C WRITE PLANT NUMERATOR COEFFICIENTS
C
  WRITE (6,70) (CO(I),I=1,NP)
  IF (NLE=0) GO TO 1
  WRITE (6,71)
C WRITE PLANT NUMERATOR ROOTS.
C
  CC 11 I=1,N
  WRITE (6,72) RTR(I),RTI(I)
11 CC CONTINUE
C
  GC TO 1
C

```

```

MAIN 490
MAIN 500
MAIN 510
MAIN 520
MAIN 530
MAIN 540
MAIN 550
MAIN 560
MAIN 570
MAIN 580
MAIN 590
MAIN 600
MAIN 610
MAIN 620
MAIN 630
MAIN 640
MAIN 650
MAIN 660
MAIN 670
MAIN 680
MAIN 690
MAIN 700
MAIN 710
MAIN 720
MAIN 730
MAIN 740
MAIN 750
MAIN 760
MAIN 770
MAIN 780
MAIN 790
MAIN 800
MAIN 810
MAIN 820
MAIN 830
MAIN 840
MAIN 850
MAIN 860
MAIN 870
MAIN 880
MAIN 890
MAIN 900
MAIN 910
MAIN 920
MAIN 930
MAIN 940
MAIN 950
MAIN 960

```

```

C LCAC PLANT DENOMINATOR ARRAY WITH CCEFFICIENTS.
C
C 12 CC 13 I=1,NP
C   E(I) = CO(I)
C 13 CC CONTINUE
C
C   WRITE (6,73)
C
C   WRITE COEFFICIENTS OF PLANT DENOMINATOR.
C
C   WRITE (6,70) (R(I),I=1,NP)
C   IF (N.LE.0) GO TO 15
C   WRITE (6,74)
C
C   WRITE ROOTS OF PLANT DENOMINATOR.
C
C   CC 14 I=1,N
C   WRITE (6,72) RTR(I),RTI(I)
C 14 CC CONTINUE
C
C 15 NTFC = N
C   N4 = NP
C   WRITE (6,62)
C
C   READ IN ASSUMED COMPENSATOR GAIN.
C
C   READ (5,64) GAINCO
C   WRITE (6,75) GAINCO
C   GC TO 1
C
C   MULTIPLY COMPENSATOR NUMERATOR COEFFICIENTS BY COMPENSATOR GAIN.
C
C 16 CC 17 I=1,NP
C   C(I) = CO(I)*GAINCO
C 17 CC CONTINUE
C
C   NCA = N
C   N1 = NP
C   WRITE (6,76)
C
C   WRITE ASSUMED COMPENSATOR NUMERATOR COEFFICIENTS
C
C   WRITE (6,70) (CO(I),I=1,NP)
C   IF (N.LE.0) GO TO 1
C   WRITE (6,77)
C
C   WRITE ROOTS OF ASSUMED COMPENSATOR NUMERATOR.
C

```

```

C      CC 18 I=1,N
C      WRITE (6,72) RTR(I),RTI(I)
C      18 CONTINUE
C      GC TO 1
C      LCAC COMPENSATOR DENCMINATOR COEFFICIENT ARRAY WITH ASSUMED STARTING
C      VALUES.
C      19 DC 20 I=1,NP
C      C(I)=CO(I)
C      20 CONTINUE
C      WRITE (6,78)
C      WRITE COEFFICIENTS OF ASSUMED COMPENSATOR DENCMINATOR POLYNOMIAL.
C      WRITE (6,70) (D(I),I=1,NP)
C      WRITE (6,75)
C      WRITE ROOTS OF ASSUMED COMPENSATOR DENOMINATOR POLYNOMIAL.
C      CC 21 I=1,N
C      WRITE (6,72) RTR(I),RTI(I)
C      21 CONTINUE
C      NCC = N
C      NZ = NP
C      WRITE (6,62)
C      READ IN PRGCRAM CONTROL DATA.
C      READ (5,80) WMIN,WMAX,NOMEG,KNOW,NBODE,NVCS,T,IZOH,NICHGL,NVINF,IC
C      18,IPLOT,ICOST
C      WRITE (6,62)
C      IF (NMINFS.EQ.0) WRITE (6,81)
C      IF (NMINFS.EQ.1) WRITE (6,82)
C      WRITE (6,62)
C      IF (IZOH.NE.1) GO TO 22
C      READ (5,64) T
C      WS = (2.0*PI)/T
C      WRITE (6,62)
C      WRITE (6,83) T,WS
C      WRITE (6,62)
C      22 KNCW = KNCW+1
C      WFRGQ(1) = WMIN
C      IF (NOMEG-2) 31,31,23
C      23 XCMEG = NOMEG

```

```

MAIN1930
MAIN1940
MAIN1950
MAIN1960
MAIN1970
MAIN1980
MAIN1990
MAIN2000
MAIN2010
MAIN2020
MAIN2030
MAIN2040
MAIN2050
MAIN2060
MAIN2070
MAIN2080
MAIN2090
MAIN2100
MAIN2110
MAIN2120
MAIN2130
MAIN2140
MAIN2150
MAIN2160
MAIN2170
MAIN2180
MAIN2190
MAIN2200
MAIN2210
MAIN2220
MAIN2230
MAIN2240
MAIN2250
MAIN2260
MAIN2270
MAIN2280
MAIN2290
MAIN2300
MAIN2310
MAIN2320
MAIN2330
MAIN2340
MAIN2350
MAIN2360
MAIN2370
MAIN2380
MAIN2390
MAIN2400

```

```

C      GC TO (29,26,24), KACH
24      CWFC = (WMAX-WMIN)/(XOMEG-1.0)
C
C      CC 25 I=2,NOMEG
25      WFREQ(I) = WFREQ(I-1)+DWFC
C
C      READ DISCRETE FREQUENCY VALUES AND DESIRED GAIN AND PHASE.
GC TO 31
26      READ (5,64) (WFREQ(I),I=1,NOMEG)
      READ (5,64) (DMAG(I),I=1,NOMEG)
      IF (ICB.NE.1) GO TO 28
C
C      IF DESIRED GAIN IS READ IN IN DB CONVERT TO A MAGNITUDE.
CC 27 I=1,NOMEG
      DWAG(I) = DMAG(I)/20.0
      DMAG(I) = 10.0**DWAG(I)
27      CCNTINUE
C
28      READ (5,64) (DQPHSD(I),I=1,NOMEG)
GC TO 31
29      Y = (ALCG10(WMAX)-ALCG10(WMIN))/(XCMEG-1.0)
      Z = 10.0**Y
C
CC 30 I=2,NOMEG
      J=I-1
30      WFREQ(I) = WFREQ(J)*Z
C
31      R = 9./13.
C
C      SET UP INITIAL GUESS, UPPER, & LOWER BOUNDS FOR SEARCH AREA.
CC 32 J=1,N1
      XS(J) = C(J)
      XL(J) = 1.E+06
      XL(J) = 0.0
      IF (MINIFS.EQ.1) XL(J)=-XU(J)
32      CCNTINUE
C
CC 33 J=1,N2
      XS(N1+J) = D(J)
      XL(N1+J) = 1.E+06
      XL(N1+J) = 0.0
33      CCNTINUE
C
      READ (5,84) NTA,NPR

```

```

      WRITE (6,62) NTA
      WRITE (6,62) ICGST
      WRITE (6,62) NAV = 0
      NV = N1+N2
      IF = 0
      CALL OPTIMIZATION ROUTINE.
      CALL BOXPLX (NV,NAV,NPR,NTA,R,XS,IP,XU,XL,YMIN,IEREXP)
      WRITE (6,62)
      WRITE (6,62) (XS(I),I=1,NV)
      WRITE (6,62) YMIN,IEREXP
      WRITE (6,62)
      LCAE COMPENSATOR NUMERATOR ARRAY WITH THE OPTIMIZED POLYNOMIAL
      COEFFICIENTS.
      EC 34 I=1,N1
      C(I) = XS(I)
      CC(1) = C(I)
      CC(1) = C(I)/C(N1)
      WRITE (6,85)
      WRITE THE OPTIMIZED COMPENSATOR NUMERATOR COEFFICIENTS IN
      LN-NORMALIZED FORM.
      WRITE (6,70) (C(I),I=1,N1)
      CC 35 I=1,N1
      CC(I) = C(I)/C(N1)
      CC(1) = C(1)
      IF (NCN.EQ.0) GO TO 37
      COMPLETE THE ROOTS OF THE OPTIMIZED COMPENSATOR NUMERATOR.
      CALL PRCD (CO,N1,RT,RT1,COF,NRCCT2,IERRT2)
      IF (IERRT2.EQ.0) GO TO 36
      CALL PREM (CO,N1,RT,RT1,COF,NRCCTB,IERRTB)
      WRITE (6,62)
      WRITE (6,63) NRCCTB,IERRTB
      WRITE (6,63)

```

```

C      WRITE (6,63)
C      WRITE (6,62)
C      WRITE (6,91)
C
C      WRITE THE ROOTS OF THE OPTIMIZED COMPENSATOR NUMERATOR.
C
C      CC 37 I=1,NCN
C      WRITE (6,52) RTR(I),RTI(I)
C      CC CONTINUE
C
C      WRITE (6,62)
C
C      LCAC COMPENSATOR DENOMINATOR ARRAY WITH THE OPTIMIZED POLYNOMIAL
C      COEFFICIENTS.
C
C      CC 38 I=1,N2
C      D(I)=XS(N1+I)
C      CC CONTINUE
C
C      WRITE (6,93)
C
C      WRITE OPTIMIZED DENOMINATOR COEFFICIENTS IN UN-NORMALIZED FCPM.
C
C      WRITE (6,70) (D(I),I=1,N2)
C
C      CC 39 I=1,N2
C      CC(I)=D(I)/D(N2)
C      CC CONTINUE
C
C      IF (NCC.EQ.0) GO TO 41
C
C      COMPLETE THE ROOTS OF THE OPTIMIZED COMPENSATOR DENOMINATOR.
C
C      CALL PRCD (CO,N2,RTR,RTI,COF,NRCCT3,IERRT3)
C      IF (IERRT3.EQ.0) GO TO 40
C      CALL PRBM (CO,N2,RTR,RTI,COF,NRCCTC,IERRTC)
C      WRITE (6,62)
C      WRITE (6,63)
C      WRITE (6,54)
C      WRITE (6,63)
C      WRITE (6,63)
C      WRITE (6,95)
C
C      WRITE THE ROOTS OF THE OPTIMIZED COMPENSATOR DENOMINATOR.
C
C      CC 41 I=1,NCN
C      WRITE (6,52) RTR(I),RTI(I)
C      CC CONTINUE
C

```

```

MAIN2890
MAIN2900
MAIN2910
MAIN2920
MAIN2930
MAIN2940
MAIN2950
MAIN2960
MAIN2970
MAIN2980
MAIN2990
MAIN3000
MAIN3010
MAIN3020
MAIN3030
MAIN3040
MAIN3050
MAIN3060
MAIN3070
MAIN3080
MAIN3090
MAIN3100
MAIN3110
MAIN3120
MAIN3130
MAIN3140
MAIN3150
MAIN3160
MAIN3170
MAIN3180
MAIN3190
MAIN3200
MAIN3210
MAIN3220
MAIN3230
MAIN3240
MAIN3250
MAIN3260
MAIN3270
MAIN3280
MAIN3290
MAIN3300
MAIN3310
MAIN3320
MAIN3330
MAIN3340
MAIN3350
MAIN3360

```



```

C      WRITE (6,62)
C
C      CCMPUTE AND WRITE THE OPTIMIZED COMPENSATOR GAIN.
C
C      GAINC = C(N1)/D(N2)
C      WRITE (6,67) NRROOT2, IERRT2, NRROOT3, IERRT3
C      CALL PMPY (COFASD,N7,A,N3,C,N1)
C      CALL PMPY (COFASD,N8,B,N4,D,N2)
C      NSD = N7-1
C      NSD = N8-1
C
C      SET UP AND EVALUATE SYSTEM OPEN LOOP FREQUENCY RESPONSE AT THE
C      DESIRED FREQUENCIES SPECIFIED USING THE OPTIMIZED COMPENSATOR
C      COEFFICIENT VALUES.
C
C      CC 52 K=1, NCMEG
C      CALL PVAL (COFASD,NSN,0.0,WFREQ(K),CNUMR,CNUMI)
C      CALL PVAL (COFASD,NSD,0.0,WFREQ(K),CENR,CENI)
C      IF (SQRT(QCENI**2+CENR**2)) 43,42,43
C
C      42 CMAG(K) = 1.E+20
C      CMAG(K) = 1.E+20
C      CI(K) = 1.E+20
C      CI(K) = 1.E+20
C      GO TO 44
C
C      43 CMAG(K) = SQRT(QNUMI**2+QNUMR**2)/SQRT(QCENI**2+CENR**2)
C      IF (IZCH.EC.1) CMAG(K)=QMAG(K)*ABS((SIN(PI*WFREQ(K)/WS)))/(PI*WFREQ
C      1(K)/WS))
C      CI(K) = (QNUMI*QDENR-QNUMR*QDENI)/(QDENR**2+QDENI**2)
C      CNUMR = ATAN2(CNUMI,CNUMR)
C      44 ANGNUR = ANGNUR*57.29578
C      ANGNUR = ANGNUR*360.0+ANGNUR
C      DELPHI = ANGNUR-PI
C      IF (IDELNU.EQ.0) GC TO 47
C      IF (DELPHI.GE.0.0.AND.IDELNU.EQ.1) GO TO 47
C      IF (DELPHI.LT.0.0.AND.IDELNU.EQ.1) GO TO 45
C      IF (DELPHI.GE.0.0.AND.IDELNU.EQ.-1) GO TC 46
C      IF (DELPHI.LT.0.0.AND.IDELNU.EQ.-1) GO TC 47
C
C      45 IF (ABS(DELPHI).LT.180.0) GC TO 47
C      DELPHI = ANGNUR+360.0-PI
C      GC TO 47
C
C      46 IF (ABS(DELPHI).LT.180.0) GC TO 47
C      DELPHI = ANGNUR-360.0-PI
C
C      47 PCHK = PCHK+DELPHI
C      PCHI = PCHI+DELPHI
C      ANGNUR = ANGNUR+PCHI*360.0

```



MAIN3850  
 MAIN3860  
 MAIN3870  
 MAIN3880  
 MAIN3890  
 MAIN3900  
 MAIN3910  
 MAIN3920  
 MAIN3930  
 MAIN3940  
 MAIN3950  
 MAIN3960  
 MAIN3970  
 MAIN3980  
 MAIN3990  
 MAIN4000  
 MAIN4010  
 MAIN4020  
 MAIN4030  
 MAIN4040  
 MAIN4050  
 MAIN4060  
 MAIN4070  
 MAIN4080  
 MAIN4090  
 MAIN4100  
 MAIN4110  
 MAIN4120  
 MAIN4130  
 MAIN4140  
 MAIN4150  
 MAIN4160  
 MAIN4170  
 MAIN4180  
 MAIN4190  
 MAIN4200  
 MAIN4210  
 MAIN4220  
 MAIN4230  
 MAIN4240  
 MAIN4250  
 MAIN4260  
 MAIN4270  
 MAIN4280  
 MAIN4290  
 MAIN4300  
 MAIN4310  
 MAIN4320

```

ANGDEC = ATAN2(QDEN1,CFENR)
ANGDEC = ANGDEC*57.29578
IF (ANGCED.LT.0.0) ANGDEC=360.0+ANGDEC
DELPSI=ANGDEC-PSI
IF (DELPSI.GE.0.0) GO TO 50
IF (DELPSI.LT.0.0) GO TO 50
IF (DELPSI.LT.0.0) AND (.ICELDE.EQ.1) GO TO 50
IF (DELPSI.LT.0.0) AND (.ICELDE.EQ.1) GO TO 48
IF (DELPSI.LT.0.0) AND (.ICELDE.EQ.-1) GO TO 45
IF (DELPSI.LT.0.0) AND (.ICELDE.EQ.-1) GO TO 50
IF (ABS(DELPSI).LT.180.0) GO TO 50
DELPSI=ANGDEC+360.0-PSI
GO TO 50
46 DELPSI=ANGDEC+360.0-PSI
GO TO 50
45 IF (ABS(DELPSI).LT.180.0) GC TO 50
IF (DELPSI=ANGDEC-360.0-PSI
50 DELPSI=PSI+DELPSI
PSI=ANGDEC
ANGDEC=ANGDEC+PSI*360.0
IF (DELPSI.GE.0.0) IDELNU=1
IF (DELPSI.LT.0.0) IDELNU=-1
IF (DELPSI.GE.0.0) IDELDE=1
IF (DELPSI.LT.0.0) IDELDE=-1
CPHSD(K)=ANGNUD-ANGDEC
CPHSD(K)=ANGNUD-ANGDEC
IF (CPHSD(K).NE.1) GO TO 51
IPS=WFREQ(K)/WS
CPHSD(K)=CPHSD(K)-(PI*WFREQ(K)/WS)*57.29578-IPS*180.0
CALCULATE THE DIFFERENCE BETWEEN THE ACTUAL ANC DESIRED RESPONSE AT
THE DISCRETE FREQUENCY VALUES ORIGINALLY SPECIFIED.
51 CPHSD(K) = (20.0*ALOG10(QMAG(K)))-(20.0*ALOG10(DMAG(K)))
52 CPHSD(K) = QPHSD(K)-DOPHSD(K)
52 CONTINUE
C
CALL SIMULA (WMIN,WMAX)
IF (NBOCCE) 55,53,55
C
C
C
PREPARE DISCRETE FREQUENCY POINT DATA FOR GRAPHING.
53 CC 54 MA=1,NOMEG
FREQ(MA) = ALOG10(WFREQ(MA))
54 CONTINUE
C
55 IF (NBOCCE.EQ.1.AND.NICOL.EQ.1) GC TO 57
CC 56 MA=1,NOMEG
ANAG(MA) = 20.0*ALOG10(QMAG(MA))
C
C

```

```

      ADMAG(MA) = 20.0*ALOG10(DMAG(MA))
56      CCNITNE
C
C
C
      CUTPLT RESULTS CN GRAPHS SPECIFIED.
57      CALL GRAPH (NBNODE,NYQST,NICPOL)
      WRITE (6,98)
      WRITE (6,99)
C
      CC 58 IJK=1,NCMEG
      WRITE (6,100) WREQ(IJK),AMAG(IJK),CPHSC(IJK),CCPSC(IJK)
      IF (IJK.EQ.1) DIFF2(IJK)
58      CCNITNE
C
C
      SYSTEM STABILITY CHECK USING ROUTH CRITERION CN CHARACTERISTIC EQA.
C
      CALL PAED (COFWRK,IWRK,COFASN,N7,COFASD,NE)
      CALL PINVRT (COFWRK,IWRK)
      CALL RCUTP (COFWRK,IWRK,ISYS)
      CALL PINVRT (COFWRK,IWRK)
      IF (ISYS.EQ.1) WRITE (6,101)
      IF (ISYS.EQ.0) WRITE (6,102)
59      STCF
C
      60      FORMAT (12A4)
      61      FORMAT (1,6X,'TITLE ---',12A4)
      62      FORMAT (1,72(' '),1)
      63      FORMAT (1,8(' *WARNING*'))
      64      FORMAT (8E10.0)
      65      FORMAT (1,6X,'UNCOMPENSATED TRANSFER FUNCTION GAIN = ',1PE13.6)
      66      FORMAT (A1,12)
      67      FORMAT (1,13,10X,'IERR1 = ',13)
      68      FORMAT (1,0,'THE FIRST CALL TO PRCD IN THE INPUT SECTION RESULTED',
1,7,'IN AN UNSUCCESSFUL DETERMINATION OF POLYNOMIAL ROOTS.',/,
2,1,'SUBROUTINE PRCD WAS CALLED TO TRY RESOLVING PROBLEM.',/,
3,A = ',13,10X,'IERR2 = ',13)
      69      FORMAT (1,13,10X,'UNCOMPENSATED TRANSFER FUNCTION NUMERATOR',/,2X,
1,1,'COEFFICIENTS IN ASCENDING POWERS OF S',/)
      70      FORMAT (1,0,5(3X,1PE13.6),1)
      71      FORMAT (1,0,6X,'UNCOMPENSATED TRANSFER FUNCTION NUMERATOR ROOTS',
1,1,'ARE:',5X,'REAL PART',8X,'IMAGINARY PART',)
      72      FORMAT (1,0,9X,1PE13.6,6X,'1PE13.6')
      73      FORMAT (1,0,6X,'UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR',/,2X,
1,1,'COEFFICIENTS IN ASCENDING POWERS OF S',/)
      74      FORMAT (1,0,6X,'UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS',
1,1,'ARE:',5X,'REAL PART',8X,'IMAGINARY PART',)
      75      FORMAT (1,0,6X,'COMPENSATOR TRANSFER FUNCTION GAIN = ',1PE13.6)

```



[illegible]

```

C+++++ SUBROUTINE PVAL
C+++++
C+++++ PURPOSE: TO EVALUATE A POLYNOMIAL A(S) WITH REAL COEFFICIENTS
C+++++ FOR WHICH THE SUBSTITUTION S=PR + JRI HAS BEEN MADE
C+++++ FOR THE COMPLEX VARIABLE.
C+++++
C+++++ CALLING SEQUENCE:
C+++++ CALL PVAL(A,NN,PR,PI,VR,VI)
C+++++
C+++++ DESCRIPTION OF PARAMETERS:
C+++++ A = VECTOR OF REAL COEFFICIENTS, ORDERED FROM LOWEST TO
C+++++ HIGHEST POWER OF THE INDEPENDENT VARIABLE.
C+++++ NN = THE ORDER OF THE POLYNOMIAL.
C+++++ PR = THE REAL PART OF THE COMPLEX VARIABLE.
C+++++ PI = THE IMAGINARY PART OF THE COMPLEX VARIABLE.
C+++++ VR = THE REAL PART OF THE RESULTANT VALUE OF THE
C+++++ POLYNOMIAL.
C+++++ VI = THE IMAGINARY PART OF THE RESULTANT VALUE OF THE
C+++++ POLYNOMIAL.
C+++++
C+++++ SUBROUTINES AND FUNCTION SUB-PROGRAMS REQUIRED:
C+++++ NONE
C+++++
C+++++ METHOD:
C+++++ EVALUATION IS PERFORMED BY MEANS OF NESTED MULTIPLICATION
C+++++
C+++++ SUBROUTINE PVAL (A,NN,PR,PI,VR,VI)
C+++++ DIMENSION A(1)
C+++++ COMPLEX S,P
C+++++ S = CMPLX(PR,PI)
C+++++ P = CMPLX(A(NN+1),0.)
C+++++ IF (NN) 4,3,1
C+++++
C+++++ DETERMINE THE COMPLEX VALUE OF THE POLYNOMIAL
C+++++
C+++++ 1 CC 2 J=1,NN
C+++++ 2 P = P*S+A(NN+1-J)
C+++++
C+++++ FIND THE REAL AND IMAGINARY PARTS AND RETURN TO CALLING PROGRAM
C+++++
C+++++ 3 VF = REAL(P)
C+++++ 4 VI = AIMAG(P)
C+++++ 5 RETURN
C+++++ 6 END

```

```

C+++++SUBROUTINE SEMBL(N,RR,RI,CF)
C+++++SUBROUTINE SEMBL
C+++++
C+++++PURPOSE TO EVALUATE THE REAL COEFFICIENTS OF A POLYNOMIAL
C+++++FROM THE REAL AND COMPLEX ROOTS OF THE POLYNOMIAL.
C+++++
C+++++CALLING SEQUENCE:
C+++++CALL SEMBL(N,RR,RI,CF)
C+++++
C+++++DESCRIPTION OF PARAMETERS:
C+++++N = INTEGER GIVING THE ORDER OF THE POLYNOMIAL. THIS
C+++++IS THE SAME AS THE NUMBER OF ROOTS.
C+++++RR = ARRAY OF DIMENSION N CONTAINING THE REAL PART CF
C+++++THE POLYNOMIAL ROOTS.
C+++++RI = ARRAY OF DIMENSION N CONTAINING THE IMAGINARY PART
C+++++OF THE POLYNOMIAL ROOTS.
C+++++CF = RESULTANT OUTPUT ARRAY CONTAINING THE REAL
C+++++COEFFICIENTS OF THE POLYNOMIAL ORDERED IN ASCENDING
C+++++POWERS OF THE INDEPENDENT VARIABLE.
C+++++
C+++++SUBROUTINES AND FUNCTION SUB-PROGRAMS REQUIRED:
C+++++NCNE
C+++++
C+++++METHOD:
C+++++THE POLYNOMIAL COEFFICIENTS ARE CALCULATED BY FORMING THE
C+++++SUMS OF THE PRODUCTS OF THE ROOTS USING COMPLEX ARITHMETIC
C+++++IN ORDER TO HANDLE ANY COMPLEX ROOTS INVOLVED.
C+++++
C+++++
C+++++SUBROUTINE SEMBL(N,RR,RI,CF)
C+++++SUBROUTINE SEMBL
C+++++THIS SUBROUTINE DETERMINES THE COEFFICIENTS CF A
C+++++POLYNOMIAL FROM ITS ROOTS
C+++++COMPLEX R(30), C(31), PRSUM
C+++++DIMENSION RR(30), RI(30), J(31), CF(31)
C+++++NN = N+1
C+++++
C+++++CC 1 I=1,N
C+++++1 R(I) = CMPLX(RR(I),RI(I))
C+++++
C+++++CF(NN) = 1.0
C+++++
C+++++CC 5 M=1,N
C+++++1 SUM = CMPLX(0.0,0.0)
C+++++
C+++++

```





```

.....
SLERCLTINE PRQD
PURPOSE
CALCULATE ALL REAL AND COMPLEX ROOTS OF A GIVEN POLYNOMIAL
WITH REAL COEFFICIENTS.
USAGE
CALL PRQD(C,IC,Q,E,POL,IR,IER)
DESCRIPTION OF PARAMETERS
C - COEFFICIENT VECTOR OF GIVEN POLYNOMIAL
  COEFFICIENTS ARE ORDERED FROM LOW TO HIGH
  THE GIVEN COEFFICIENT VECTOR GETS DIVIDED BY THE
  LAST NONZERO TERM
IC - DIMENSION OF VECTOR C
Q - WORKING STORAGE OF DIMENSION IC OF REAL PARTS OF ROOTS
E - ON RETURN Q CONTAINS DIMENSION IC OF COMPLEX PARTS OF ROOTS
PCL - ON RETURN POL CONTAINS THE COEFFICIENTS OF THE
      POLYNOMIAL WITH CALCULATED ROOTS
IR - THIS RESULTS ARE ORDERED FROM LOW TO HIGH
      NUMBER OF CALCULATED ROOTS
IER - NORMALLY IER IS EQUAL TO DIMENSION IC MINUS ONE
      RESULTING ERROR PARAMETER. SEE REMARKS
REMARKS
THE REAL PART OF THE ROOTS IS STORED IN Q(1) UP TO Q(IR).
THE CORRESPONDING COMPLEX PARTS ARE STORED IN E(1) UP TO E(IR).
IER = 0 MEANS NO ERROR OCCURRED WITH FEASIBLE TOLERANCE
IER = 1 MEANS POLYNOMIAL IS DEGENERATE (CONSTANT OR ZERO)
IER = 2 MEANS SUBROUTINE WAS ABANDONED DUE TO ZERO DIVISOR
IER = 3 MEANS THERE EXISTED COEFFICIENTS OF ZERO DEGREE
IER = 4 MEANS CALCULATED COEFFICIENT VECTOR REVEALS PCCR
      ACCURACY OF THE CALCULATED ROOTS
      THE CORRECTED COEFFICIENT VECTOR HAS LESS THAN
      3 CORRECT DIGITS.
THE FINAL COMPARISON BETWEEN GIVEN AND CALCULATED
COEFFICIENT VECTOR IS PERFORMED ONLY IF ALL ROOTS HAVE BEEN
CALCULATED.
THE MAXIMAL RELATIVE ERROR OF THE COEFFICIENT VECTOR IS
RECORDED IN Q(IR+1).

```

CC



SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED  
NCNE

METHCD ROOTS OF THE POLYNOMIAL ARE CALCULATED BY MEANS OF  
THE QUOTIENT-DIFFERENCE ALGORITHM WITH DISPLACEMENT.  
REFERENCE  
H. RUTISHAUSER, DER QUOTIENTEN-DIFFERENZEN-ALGORITHMUS,  
BIRKHAUSER, BASEL/STUTTGART, 1957.

.....  
SUBROUTINE PRQD (C,IC,C,E,PCL,IF,IER)

DIMENSIONED DUMMY VARIABLES

DIMENSION E(1), Q(1), C(1), POL(1)

NORMALIZATION OF GIVEN POLYNOMIAL

TEST OF DIMENSION

IR CCNTAINS INDEX OF HIGHEST CCEFFICIENT

IER = 0

IR = IC

EPS = 1.E-6

TOL = 1.E-3

LIMIT = 10\*IC

KCOUNT = 0

1 IF ((IR-1) 79,79,2

2 IF (C(IR)) 4,3,4

3 IF = IR-1

GC TO 1

REARRANGEMENT OF GIVEN POLYNOMIAL

EXTRACTION OF ZERO ROOTS

4 Q = 1./C(IR)

IENC = IR-1

ISTA = 1

NSAV = IR+1

JREC = 1

C(J)=1./C(IR-1)/C(IR)

C(J+1)=C(J)/C(IR)

WHERE J IS THE INDEX OF THE LOWEST NONZERO COEFFICIENT

CC 5 I=1,IQ

J = NSAV-I

PRQD 450  
PRQD 500  
PRQD 510  
PRQD 520  
PRQD 530  
PRQD 540  
PRQD 550  
PRQD 560  
PRQD 570  
PRQD 580  
PRQD 590  
PRQD 600  
PRQD 610  
PRQD 620  
PRQD 630  
PRQD 640  
PRQD 650  
PRQD 660  
PRQD 670  
PRQD 680  
PRQD 690  
PRQD 700  
PRQD 710  
PRQD 720  
PRQD 730  
PRQD 740  
PRQD 750  
PRQD 760  
PRQD 770  
PRQD 780  
PRQD 790  
PRQD 800  
PRQD 810  
PRQD 820  
PRQD 830  
PRQD 840  
PRQD 850  
PRQD 860  
PRQD 870  
PRQD 880  
PRQD 890  
PRQD 900  
PRQD 910  
PRQD 920  
PRQD 930  
PRQD 940  
PRQD 950



```

C C      EUCLIDEAN ALGORITHM
16 C      19 K=1, IEND
   C      E(K+1) = E(K+1)/E(I)
   C      E(K+1) = E(K+1)/-Q(K+1)
   C      IF (K-1) 18,17,18
C C
17 C      TEST FOR SMALL DIVISOR
18 C      IF (ABS(Q(I+1))-PCL(I+1)) 80,80,15
   C      E(K+1) = Q(K+1)/Q(I+1)
   C      PCL(K+1) = POL(K+1)/ABS(Q(I+1))
   C      E(K) = C(K+1)-E(K)
   C      CC CONTINUE
C C
20 C      G(IR) = -C(IR)
C C
C C      THE DISPLACEMENT EXPT IS SET TO 0 AUTOMATICALLY.
C C      E(ISTA)=0,Q(ISTA+1),...E(NSAV-1),C(NSAV),E(NSAV)=0.,
C C      FORM A DIAGONAL OF THE QD-ARRAY.
C C      INITIALIZATION CF BOUNDARY VALUES
21 C      E(ISTA) = 0.
   C      NRAW = NSAV-1
   C      E(NRAW+1) = 0.
C C
C C      TEST FOR LINEAR OR CONSTANT FACTOR
   C      NRAW-ISTA IS DEGREE-1
   C      IF (NRAW-ISTA) 24,23,31
C C
C C      LINEAR FACTOR
23 C      Q(ISTA+1) = Q(ISTA+1)+EXPT
   C      E(ISTA+1) = 0.
C C
C C      TEST FOR UNFACTORED COMMON DIVISOR
24 C      E(ISTA) = E$AV
   C      IF (IR-NSAV) 60,60,25
C C
C C      INITIALIZE QD-ALGORITHM FOR COMMON DIVISOR
25 C      ISTA = NSAV
   C      E$AV = E(ISTA)
   C      GC TO 10
C C
C C      COMPUATION OF RCOT PAIR
26 C      P = P+EXPT
C C
C C      TEST FOR REALITY
   C      IF (D) 27,28,28
C C
C C      COMPLEX ROOT PAIR

```

```

27 C(NRAN) = P
   C(NRAN+1) = P
   E(NRAN) = T
   E(NRAN+1) = -T
   GC TO 25
C
28 REAL ROOT PAIR
   C(NRAN) = P-T
   C(NRAN+1) = P+T
   E(NRAN) = 0.
C
29 REDUCTION OF DEGREE BY 2 (DEFLATION)
   NRAN = NRAN-2
   GC TO 22
C
30 COMPLETION OF REAL ROOT
   C(NRAN+1) = EXPT+P
C
31 REDUCTION OF DEGREE BY 1 (DEFLATION)
   NRAN = NRAN-1
   GC TO 22
C
32 START QD-ITERATION
   JECC = ISTA+1
   JENC = NRAN-1
   TEPS = EPS
   TCELT = 1.E-2
   KCLMT = KOUNT+1
   P = C(NRAN+1)
   R = ABS(E(NRAN))
C
33 TEST FOR CONVERGENCE
   IF (R-TEPS) 30,30,33
   S = ABS(E(JEND))
C
34 IS THERE A REAL ROOT NEXT
   IF (S-R) 38,38,34
C
35 IS DISPLACEMENT SMALL ENOUGH
   IF (R-TCELT) 36,35,35
36 P = 0.
37 C = P
C
38 J=JBEG,NRAN
   C(J) = C(J)+E(J)-E(J-1)-O
C
39 TEST FOR SMALL DIVISOR
   IF (ABS(Q(J))-POL(J)) 81,81,37

```

```

PRGD1930
PRGD1940
PRGD1950
PRGD1960
PRGD1970
PRGD1980
PRGD1990
PRGD2000
PRGD2010
PRGD2020
PRGD2030
PRGD2040
PRGD2050
PRGD2060
PRGD2070
PRGD2080
PRGD2090
PRGD2100
PRGD2110
PRGD2120
PRGD2130
PRGD2140
PRGD2150
PRGD2160
PRGD2170
PRGD2180
PRGD2190
PRGD2200
PRGD2210
PRGD2220
PRGD2230
PRGD2240
PRGD2250
PRGD2260
PRGD2270
PRGD2280
PRGD2290
PRGD2300
PRGD2310
PRGD2320
PRGD2330
PRGD2340
PRGD2350
PRGD2360
PRGD2370
PRGD2380
PRGD2390
PRGD2400

```



```

C = P*P-(V-U)*(Q-U*T-Q*W*(1.+T)/C(JEND))
T = SQRT(ABS(Q))
GC TO 26

      TEST FOR SMALL DIVISOR
45 IF (ABS(Q)-POL(J+1)) 46,46,50
50 W = U*C/V
    T = T*(V/Q)**2
    C(J) = V+W-E(J-1)
    U = 0.
51 IF (J-NRAN) 51,52,52
52 U = Q(J+2)*E(J+1)/(C*(1.+T))
    V = O+U-W

      TEST FOR SMALL DIVISOR
53 IF (ABS(Q(J))-POL(J)) 81,81,53
    E(J) = W*V*(1.+T)/Q(J)
    C(NRAN+1) = V-E(NRAN)
54 E(J)PT = EXPT+P
    TEPS = TEPS*.1
    TDEL = TDEL*.1
    IF (KCOUNT-LIMIT) 32,55,55

      NO CONVERGENCE WITH FEASIBLE TOLERANCE
      ERROR RETURN IN CASE OF UNSATISFACTORY CONVERGENCE
55 IER = 1

      REARRANGE CALCULATED ROOTS
56 IF (C(1)-NSAV-NRAN-1)
    E(ISTA) = ESAV
    IF (IEND) 59,59,57

57 DO 58 I=1,IEND
    J = ISTA+I
    K = NRAN+1+I
    E(J) = E(K)
58 C(J) = C(K)

59 IF = ISTA+IEND

60 IF = IR-1
    IF (IR) 78,78,61

      REARRANGE CALCULATED ROOTS
61 DO 62 I=1,IR

```

```

PRQD2850
PRQD2900
PRQD2910
PRQD2920
PRQD2930
PRQD2940
PRQD2950
PRQD2960
PRQD2970
PRQD2980
PRQD2990
PRQD3000
PRQD3010
PRQD3020
PRQD3030
PRQD3040
PRQD3050
PRQD3060
PRQD3070
PRQD3080
PRQD3090
PRQD3100
PRQD3110
PRQD3120
PRQD3130
PRQD3140
PRQD3150
PRQD3160
PRQD3170
PRQD3180
PRQD3190
PRQD3200
PRQD3210
PRQD3220
PRQD3230
PRQD3240
PRQD3250
PRQD3260
PRQD3270
PRQD3280
PRQD3290
PRQD3300
PRQD3310
PRQD3320
PRQD3330
PRQD3340
PRQD3350

```

```

C      S(I) = S(I+1)
C      E(I) = E(I+1)
C
C      CALCULATE COEFFICIENT VECTOR FROM ROOTS
C
C      FCL(IR+1) = 1.
C      JENC = IR-1
C      JEEG = 1
C
C      CC 69 J=1,IR
C      ISTA = IR+1-J
C      C = 0.
C      F = Q(I,ISTA)
C      T = E(I,ISTA)
C      IF (T) 65,63,65
C
C      MULTIPLY WITH LINEAR FACTOR
C
C      CC 64 I=ISTA,IR
C      FCL(I) = C-P*POL(I+1)
C      C = POL(I+1)
C
C      GC TO 65
C      JREG = 2, JBEG
C      FCL(ISTA) = 0.
C      GC TO 65
C
C      MULTIPLY WITH QUADRATIC FACTOR
C
C      67 JREG = 1
C      L = P*P*T*T
C      F = P*P
C
C      CC 68 I=ISTA,IEND
C      FCL(I) = C-P*POL(I+1)+U*POL(I+2)
C      C = PCL(I+1)
C
C      FCL(IR) = C-P
C      CC CONTINUE
C
C      IF (IER) 78,70,78
C
C      COMPARISON OF COEFFICIENT VECTORS, IE. TEST OF ACCURACY
C      7C F = 0.
C
C      CC 75 I=1,IR
C      IF (C(I)) 72,71,72
C      71 Q = ABS(POL(I))

```

```

PRQD3850
PRQD3860
PRQD3870
PRQD3880
PRQD3890
PRQD3900
PRQD3910
PRQD3920
PRQD3930
PRQD3940
PRQD3950
PRQD3960
PRQD3970
PRQD3980
PRQD3990
PRQD4000
PRQD4010
PRQD4020
PRQD4030
PRQD4040
PRQD4050
PRQD4060
PRQD4070
PRQD4080
PRQD4090
PRQD4100
PRQD4110

```

```

C GC TC 73
  72 C = ABS((POL(I)-C(I))/C(I))
  73 IF (P-C) 74,75,75
  74 P = 0
  75 CONTINUE
C
  76 IF (P-TCL) 77,76,76
  76 IER = -1
  77 C((IR+1)) = P
  77 C((IP+1)) = 0.
  78 RETURN
C
  79 ERROR RETURNS
  79 ERROR RETURN FOR POLYNOMIALS OF DEGREE LESS THAN 1
  79 IER = 2
  79 IR = 0
  79 RETURN
C
  80 ERROR RETURN IF THERE EXISTS NO S-FRACTION
  80 IER = 4
  80 IR = ISIA
  80 GC TC 60
C
  81 ERROR RETURN IN CASE OF INSTABLE QC-ALGORITHM
  81 IER = 3
  81 GC TC 56
  81 ENC

```



```

.....
SUBROUTINE PRBM
PURPOSE
  TO CALCULATE ALL REAL AND COMPLEX ROOTS OF A GIVEN
  POLYNOMIAL WITH REAL COEFFICIENTS.
USAGE
  CALL PRBM (C,IC,RR,RC,POL,IF,IER)
DESCRIPTION OF PARAMETERS CONTAINING THE COEFFICIENTS OF THE
  GIVEN POLYNOMIAL. COEFFICIENTS ARE ORDERED FROM
  LOW TO HIGH. ON RETURN COEFFICIENTS ARE DIVIDED
  BY THE LAST NONZERO TERM. RC, AND PCL
  DIMENSION VECTOR OF REAL PARTS OF THE ROOTS.
  RR - RESULTANT VECTOR OF COMPLEX PARTS OF THE ROOTS.
  PCL - RESULTANT VECTOR OF COEFFICIENTS OF THE POLYNOMIAL
  WITH CALCULATED ROOTS. (SEE REMARK 4)
  FROM LOW TO HIGH (SEE REMARK 4)
  IR - OUTPUT VALUE SPECIFYING THE NUMBER OF CALCULATED
  ROOTS. NORMALLY IR IS EQUAL TO IC-1.
  IER - RESULTANT ERROR PARAMETER CODED AS FOLLOWS
      IER=0 - NO ERROR
      IER=1 - SUBROUTINE PQFB RECORDS POOR CONVERGENCE
        AT SOME QUADRATIC FACTORIZATION WITHIN
        ITERATION STEPS
      IER=2 - POLYNOMIAL IS DEGENERATE, I.E. ZERO CR
        OR CONSTANT, IN NORMALIZATION OF GIVEN
        POLYNOMIAL
      IER=3 - THE SUBROUTINE IS BYPASSED DUE TO
        SUCCESSIVE ZERO DIVISORS OR OVERFLOWS
        IN QUADRATIC FACTORIZATION. CUE TO
        COMPLETELY UNSUCCESSFUL FACTORIZATION
        CALCULATED COEFFICIENT VECTOR PCL IS LESS
        THAN THREE CORRECT SIGNIFICANT DIGITS
        THIS REVEALS POOR ACCURACY OF CALCULATED
        ROOTS.
REMARKS
  (1) REAL PARTS OF THE ROOTS ARE STORED IN RR(1) UP TO RC (IR)
      AND CORRESPONDING COMPLEX PARTS IN RC(1) UP TO RC (IR).
  (2) ERROR MESSAGE IER=1 INDICATES POOR CONVERGENCE WITHIN
      50 ITERATION STEPS AT SOME QUADRATIC FACTORIZATION
      PERFORMED BY SUBROUTINE PQFB.

```

```

PRBM 10
PRBM 20
PRBM 30
PRBM 40
PRBM 50
PRBM 60
PRBM 70
PRBM 80
PRBM 90
PRBM 100
PRBM 110
PRBM 120
PRBM 130
PRBM 140
PRBM 150
PRBM 160
PRBM 170
PRBM 180
PRBM 190
PRBM 200
PRBM 210
PRBM 220
PRBM 230
PRBM 240
PRBM 250
PRBM 260
PRBM 270
PRBM 280
PRBM 290
PRBM 300
PRBM 310
PRBM 320
PRBM 330
PRBM 340
PRBM 350
PRBM 360
PRBM 370
PRBM 380
PRBM 390
PRBM 400
PRBM 410
PRBM 420
PRBM 430
PRBM 440
PRBM 450
PRBM 460
PRBM 470
PRBM 480

```

```

450 PRBM
500 PRBM
510 PRBM
520 PRBM
530 PRBM
540 PRBM
550 PRBM
560 PRBM
570 PRBM
580 PRBM
590 PRBM
600 PRBM
610 PRBM
620 PRBM
630 PRBM
640 PRBM
650 PRBM
660 PRBM
670 PRBM
680 PRBM
690 PRBM
700 PRBM
710 PRBM
720 PRBM
730 PRBM
740 PRBM
750 PRBM
760 PRBM
770 PRBM
780 PRBM
790 PRBM
800 PRBM
810 PRBM
820 PRBM
830 PRBM
840 PRBM
850 PRBM
860 PRBM
870 PRBM
880 PRBM
890 PRBM
900 PRBM
910 PRBM
920 PRBM
930 PRBM
940 PRBM
950 PRBM
960 PRBM

```

- (3) NO ACTION BESIDES ERROR MESSAGE IER=2 IN CASE CF A ZERO OR CONSTANT POLYNOMIAL. THE SAME ERROR MESSAGE IS GIVEN IN CASE OF AN OVERFLOW. IN NORMALIZATION OF GIVEN POLYNOMIAL.
- (4) ERROR MESSAGE IER=3 INDICATES SUCCESSIVE ZERO DIVISORS OR OVERFLOW. CF COMPLETELY UNSUCCESSFUL BY ANY QUADRATIC FACTORIZATION CASE CALCULATED BY SUBROUTINE PCFB. IN THIS CASE CALCULATION IS BYPASSED. IR RECORDS THE NUMBER OF CALCULATED ROOTS. CF THE POL(1),...POL(J-1) ARE THE COEFFICIENTS OF THE REMAINDER IN VECTOR C (NORMALLY J-1). THE ACTUAL NUMBER OF COEFFICIENTS IN VECTOR C (NORMALLY J-1) IS LESS THAN THREE IF CALCULATED COEFFICIENTS THROUGH ALL QUADRATIC FACTORIZATIONS SHOWED UNSUCCESSFUL. THE ERROR MESSAGE IER=1 IS GIVEN. WHEN GIVEN AND CALCULATED THE FINAL COMPONENT IS PERFORMED ONLY IF ALL ROOTS HAVE BEEN CALCULATED. IN THIS CASE THE NUMBER OF ROOTS IR IS EQUAL TO THE ACTUAL DEGREE OF THE POLYNOMIAL (NORMALLY IR=JC-1). THE MAXIMAL RELATIVE ERROR OF THE COEFFICIENT VECTOR IS RECORDED IN RL(IR+1).
- (5) SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
- (6) SUBROUTINE PCFB

QUADRATIC FACTORIZATION OF A POLYNOMIAL  
BY BAIRSTOW ITERATION.

METHOD

THE ROOTS OF THE POLYNOMIAL ARE CALCULATED BY MEANS OF SUCCESSIVE QUADRATIC FACTORIZATION PERFORMED BY BAIRSTOW ITERATION. X\*2 IS USED AS INITIAL GUESSES FOR THE FIRST QUADRATIC FACTOR. AND FURTHER GUESSES FOR THE SECOND QUADRATIC FACTOR. INITIAL THE COEFFICIENT VECTOR IS COMPUTED AND ALL ROOTS OF THE GIVEN CNE. THE EVALUATION OF ALL ROOTS WITH THE GIVEN CNE. THE EVALUATION OF THE ZEROS OF THE POLYNOMIALS (PART ONE AND TWO), NUMERISCHE MATHEMATIK, VOL.1 (1959), PP.150-180.

SUBROUTINE PRBM (C,IC,RR,RC,POL,IF,IER)

DIMENSION C(1), RR(1), RC(1), POL(1), Q(4)

TEST ON LEADING ZERO COEFFICIENTS

```

EFS = 1.E-3
LIM = 50
IH = IC+1

```

```

1  IR = IR-1 42,42,2
2  IF (C(IR)) 3,1,3
3  WCRK UP ZERO ROOTS AND NORMALIZE REMAINING POLYNOMIAL
4  IER = 0
5  J = IR
6  L = 0
7  A = C(IR)
8  CC 8 I=1,IR
9  IF (L) 4,4,7
10 IF (C(I)) 6,5,6
11 RR(I) = 0.
12 FC(I) = 0.
13 J = J-1
14 CC TO 8
15 L = L+1
16 J = 0
17 J = J+1
18 C(I) = C(I)/A
19 FC(J) = C(I)
20 CALL OVERFL(N)
21 IF (N-2) 42,8,8
22 CC CONTINUE
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

```

C C C C
GC TO 34
THIS IS BRANCH TO COMPARISON OF COEFFICIENT VECTORS C AND PCL
DEGREE OF RESTPOLYNOMIAL IS GREATER THAN ONE
14 CC 22 L=1,10
15 N=1
15 C(1)=C1
15 C(2)=C2
15 CALL PCF8 (POL,J,G,LIM,I)
16 IF (C1) 16,24,23
17 IF (C2) 18,21,18
18 GC TO (19,20,19,21), N
19 C1=-C1
20 N=N+1
20 C2=-C2
21 N=N+1
21 C1=1.-C1
22 C2=1.-C2
C C C
      ERROR EXIT DUE TO UNSATISFACTORY RESULTS OF FACTORIZATION
      IER=3
      IR=IR-J
      RETURN
C C
      WORK UP RESULTS OF QUADRATIC FACTORIZATION
23 IER=1
24 C1=C(1)
24 C2=C(2)
C C
      PERFORM DIVISION OF FACTORIZED POLYNOMIAL BY QUADRATIC FACTOR
      B=0.
      A=0.
      I=J
      I=-Q1*B-Q2*A+POL(I)
25 PCL(I)=B
      E=A
      A=H
      I=I-1
26 PCL(I-2)=B
      PCL(I-2)=A
C C
      MULTIPLY POLYNOMIAL WITH CALCULATED ROOTS BY QUADRATIC FACTOR

```

```

PRB1450
PRB1460
PRB1470
PRB1480
PRB1490
PRB1500
PRB1510
PRB1520
PRB1530
PRB1540
PRB1550
PRB1560
PRB1570
PRB1580
PRB1590
PRB1600
PRB1610
PRB1620
PRB1630
PRB1640
PRB1650
PRB1660
PRB1670
PRB1680
PRB1690
PRB1700
PRB1710
PRB1720
PRB1730
PRB1740
PRB1750
PRB1760
PRB1770
PRB1780
PRB1790
PRB1800
PRB1810
PRB1820
PRB1830
PRB1840
PRB1850
PRB1860
PRB1870
PRB1880
PRB1890
PRB1900
PRB1910
PRB1920

```

```

PRBM15330
PRBM15400
PRBM15500
PRBM15600
PRBM15700
PRBM15800
PRBM15900
PRBM16000
PRBM20010
PRBM20020
PRBM20030
PRBM20040
PRBM20050
PRBM20060
PRBM20070
PRBM20080
PRBM20090
PRBM21100
PRBM21110
PRBM21120
PRBM21130
PRBM21140
PRBM21150
PRBM21160
PRBM21170
PRBM21180
PRBM21190
PRBM22200
PRBM22210
PRBM22220
PRBM22230
PRBM22240
PRBM22250
PRBM22260
PRBM22270
PRBM22280
PRBM22290
PRBM22300
PRBM22310
PRBM22320
PRBM22330
PRBM22340
PRBM22350
PRBM22360
PRBM22370
PRBM22380
PRBM22390
PRBM22400

```

```

C
L = IR-1
IF (J-L) 27,27,29
27 CC 28 I=J,L
28 PCL(I-1) = PCL(I-1)+POL(I)*C2+PCL(I+1)*C1
C
29 PCL(L) = PCL(L)+PCL(L+1)*C2+C1
30 PCL(IR) = PCL(IR)+C2
C
C      CALCULATE ROOT-FAIR FROM QUADRATIC FACTOR X*X+C2*X+Q1
F = -.5*C2
A = H*F-Q1
B = SQRT(ABS(A))
IF (A) 30,30,31
30 RCL(IST) = B
IRCL(IST) = IST+1
IRCL(IST) = -B
GO TO 32
31 B = H+SIGN(B,H)
RCL(IST) = Q1/B
IRCL(IST) = 0.
IRCL(IST) = IST+1
RCL(IST) = 0.
32 J = J-2
CC TC 9
C
C      SHIFT BACK ELEMENTS OF POL BY 1 AND COMPARE VECTORS PCL AND C
33 IR = IR-1
34 A = 0.
C
CC 38 I=1, IR
C1 = C(I)
C2 = PCL(I+1)
IF (Q1) 35,36,35
C2 = (C1-C2)/Q1
35 IF (C2-A) 38,38,37
36 A = Q2
37 CC CONTINUE
C
I = IR+1
PCL(I) = 1.
RCL(I) = A

```

```

PRBM2410
PRBM2420
PRBM2430
PRBM2440
PRBM2450
PRBM2460
PRBM2470
PRBM2480
PRBM2490
PRBM2500
PRBM2510
PRBM2520
PRBM2530
PRBM2540

```

```

C      R(1) = 0.
C      IF (IER) 35,39,41
C      IF (A-EP5) 41,41,40
35      WARNING DUE TO PCOR ACCURACY OF CALCULATED COEFFICIENT VECTOR
40      IER = -1
41      RETURN
C      ERROR EXIT DUE TO DEGENERATE POLYNOMIAL OR OVERFLOW IN
C      NORMALIZATION
42      IER = 2
C      IF (IER) 0
C      RETURN
C      END

```





```

(5) NORM OF THE MODIFIED LINEAR REMAINDER,
    FOR COMPLETE DETAIL SEE THE DOCUMENTATION FOR
    SUBROUTINES PQFB AND CPCFB.
SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
NCNE
METHOD
COMPUTATION IS BASED ON BAIRSTOW'S ITERATIVE METHOD. (SEE
WILKINSON, J.H., THE EVALUATION OF THE ZEROS OF ILL-CO-
NITIONED POLYNOMIALS (PART ONE AND TWO), NUMERISCHE MATHE-
MATIK VOL. 1 (1959) PP. 150-180, OR FIDELL, D.B., YCRK/
INTRODUCTION TO NUMERICAL ANALYSIS,
TCRONT/LONDON, 1956, PP. 472-476.)
.....
SUBROUTINE PQFB (C,IC,CLIM,IER)
.....
DIMENSION C(1), Q(1)
TEST ON LEADING ZERO COEFFICIENTS
IER = 0
J = IC+1
1 J = J-1
2 IF (C(J)) 3,1,3
3 A = C(J)
4 IF (A-1.) 4,6,4
5 CC = I=1,J
C(I) = C(I)/A
CALL OVERFL(N)
IF (N-2) 40,5,5
5 CONTINUE
TEST ON NECESSITY OF BAIRSTOW ITERATION
6 IF (J-3) 41,38,7
PREPARE BAIRSTOW ITERATION
7 EPS = 1.E-6
EPS1 = 1.E-3
L = 0
LL = 0

```



```

1  = G(1)
2  = Q(1)
3  = C(1)
4  = ABS(AA)
5  = ABS(BB)
6  = (CE-CA) 8,9,10
7  = CB+CB
8  = CB/CA
9  = 1
10 = 1
11 = 1
12 = 1
13 = 1
14 = 1
15 = 1
16 = 1
17 = 1
18 = 1
19 = 1
20 = 1
21 = 1
22 = 1
23 = 1
24 = 1
25 = 1
26 = 1
27 = 1
28 = 1
29 = 1
30 = 1
31 = 1
32 = 1
33 = 1
34 = 1
35 = 1
36 = 1
37 = 1
38 = 1
39 = 1
40 = 1
41 = 1
42 = 1
43 = 1
44 = 1

```

START BAIRSTOW ITERATION  
PREPARE NESTED MULTIPLICATION

```

1  = Q(1)
2  = C(1)
3  = ABS(AA)
4  = ABS(BB)
5  = (CE-CA) 8,9,10
6  = CB+CB
7  = CB/CA
8  = 1
9  = 1
10 = 1
11 = 1
12 = 1
13 = 1
14 = 1
15 = 1
16 = 1
17 = 1
18 = 1
19 = 1
20 = 1
21 = 1
22 = 1
23 = 1
24 = 1
25 = 1
26 = 1
27 = 1
28 = 1
29 = 1
30 = 1
31 = 1
32 = 1
33 = 1
34 = 1
35 = 1
36 = 1
37 = 1
38 = 1
39 = 1
40 = 1
41 = 1
42 = 1
43 = 1
44 = 1

```

START NESTED MULTIPLICATION

```

1  = Q(1)
2  = C(1)
3  = ABS(AA)
4  = ABS(BB)
5  = (CE-CA) 8,9,10
6  = CB+CB
7  = CB/CA
8  = 1
9  = 1
10 = 1
11 = 1
12 = 1
13 = 1
14 = 1
15 = 1
16 = 1
17 = 1
18 = 1
19 = 1
20 = 1
21 = 1
22 = 1
23 = 1
24 = 1
25 = 1
26 = 1
27 = 1
28 = 1
29 = 1
30 = 1
31 = 1
32 = 1
33 = 1
34 = 1
35 = 1
36 = 1
37 = 1
38 = 1
39 = 1
40 = 1
41 = 1
42 = 1
43 = 1
44 = 1

```

```

1  A1 = H
2  TO 13
3  END CF NESTED MULTIPLICATION
4  C
5  C
6  C
7  TEST ON SATISFACTORY ACCURACY
8  IF (CA*ABS(A)+CB*ABS(B))
9  18 IF (LL) 19,19,39
10 19 L = L+1
11 20 IF (ABS(A)-EPS*ABS(C(1))) 20,20,21
12 21 IF (ABS(B)-EPS*ABS(C(2))) 21,21,22
13 22 IF (H-CC) 22,22,23
14 23 IF AA = A
15 24 BE = B
16 25 CC1 = C1
17 26 CC2 = C2
18 27 TEST ON LAST ITERATION STEP
19 28 IF (L-LIM) 28,28,24
20 29 C
21 30 C
22 31 C
23 32 TEST CN RESTART OF BAIRSTOW ITERATION WITH ZERO INITIAL GUESS
24 33 IF (H-CQ) 43,43,25
25 34 IF (G(1)) 27,26,27
26 35 IF (G(2)) 27,42,27
27 36 C(1) = 0.
28 37 C(2) = 0.
29 38 TO 7
30 39 C
31 40 C
32 41 PERFORM ITERATION STEP
33 42 IF (H) 42,42,25
34 43 IF (H) 42,42,25
35 44 A1 = A1/H
36 45 B1 = B1/H
37 46 C1 = C1/H
38 47 IF (A1-C1-H-B1*B1)
39 48 IF (H) 30,42,30
40 49 A = A/H
41 50 B = B/H
42 51 H = (B*A1-A*B1)/H
43 52 H = (A*C1-B*B1)/H
44 53 C1 = (A1+H)
45 54 C2 = (B2+H)
46 55 END CF ITERATION STEP
47 56 C
48 57 C
49 58 C
50 59 TEST ON SATISFACTORY RELATIVE ERROR OF ITERATED VALUES
51 60 IF (ABS(HH)-EPS*ABS(Q1)) 31,31,33
52 61 C
53 62 C
54 63 C
55 64 C
56 65 C
57 66 C
58 67 C
59 68 C
60 69 C
61 70 C
62 71 C
63 72 C
64 73 C
65 74 C
66 75 C
67 76 C
68 77 C
69 78 C
70 79 C
71 80 C
72 81 C
73 82 C
74 83 C
75 84 C
76 85 C
77 86 C
78 87 C
79 88 C
80 89 C
81 90 C
82 91 C
83 92 C
84 93 C
85 94 C
86 95 C
87 96 C
88 97 C
89 98 C
90 99 C
91 100 C
92 101 C
93 102 C
94 103 C
95 104 C
96 105 C
97 106 C
98 107 C
99 108 C
100 109 C
101 110 C
102 111 C
103 112 C
104 113 C
105 114 C
106 115 C
107 116 C
108 117 C
109 118 C
110 119 C
111 120 C
112 121 C
113 122 C
114 123 C
115 124 C
116 125 C
117 126 C
118 127 C
119 128 C
120 129 C
121 130 C
122 131 C
123 132 C
124 133 C
125 134 C
126 135 C
127 136 C
128 137 C
129 138 C
130 139 C
131 140 C
132 141 C
133 142 C
134 143 C
135 144 C
136 145 C
137 146 C
138 147 C
139 148 C
140 149 C
141 150 C
142 151 C
143 152 C
144 153 C
145 154 C
146 155 C
147 156 C
148 157 C
149 158 C
150 159 C
151 160 C
152 161 C
153 162 C
154 163 C
155 164 C
156 165 C
157 166 C
158 167 C
159 168 C
160 169 C
161 170 C
162 171 C
163 172 C
164 173 C
165 174 C
166 175 C
167 176 C
168 177 C
169 178 C
170 179 C
171 180 C
172 181 C
173 182 C
174 183 C
175 184 C
176 185 C
177 186 C
178 187 C
179 188 C
180 189 C
181 190 C
182 191 C
183 192 C
184 193 C
185 194 C
186 195 C
187 196 C
188 197 C
189 198 C
190 199 C
191 200 C
192 201 C
193 202 C
194 203 C
195 204 C
196 205 C
197 206 C
198 207 C
199 208 C
200 209 C
201 210 C
202 211 C
203 212 C
204 213 C
205 214 C
206 215 C
207 216 C
208 217 C
209 218 C
210 219 C
211 220 C
212 221 C
213 222 C
214 223 C
215 224 C
216 225 C
217 226 C
218 227 C
219 228 C
220 229 C
221 230 C
222 231 C
223 232 C
224 233 C
225 234 C
226 235 C
227 236 C
228 237 C
229 238 C
230 239 C
231 240 C
232 241 C
233 242 C
234 243 C
235 244 C
236 245 C
237 246 C
238 247 C
239 248 C
240 249 C
241 250 C
242 251 C
243 252 C
244 253 C
245 254 C
246 255 C
247 256 C
248 257 C
249 258 C
250 259 C
251 260 C
252 261 C
253 262 C
254 263 C
255 264 C
256 265 C
257 266 C
258 267 C
259 268 C
260 269 C
261 270 C
262 271 C
263 272 C
264 273 C
265 274 C
266 275 C
267 276 C
268 277 C
269 278 C
270 279 C
271 280 C
272 281 C
273 282 C
274 283 C
275 284 C
276 285 C
277 286 C
278 287 C
279 288 C
280 289 C
281 290 C
282 291 C
283 292 C
284 293 C
285 294 C
286 295 C
287 296 C
288 297 C
289 298 C
290 299 C
291 300 C
292 301 C
293 302 C
294 303 C
295 304 C
296 305 C
297 306 C
298 307 C
299 308 C
300 309 C
301 310 C
302 311 C
303 312 C
304 313 C
305 314 C
306 315 C
307 316 C
308 317 C
309 318 C
310 319 C
311 320 C
312 321 C
313 322 C
314 323 C
315 324 C
316 325 C
317 326 C
318 327 C
319 328 C
320 329 C
321 330 C
322 331 C
323 332 C
324 333 C
325 334 C
326 335 C
327 336 C
328 337 C
329 338 C
330 339 C
331 340 C
332 341 C
333 342 C
334 343 C
335 344 C
336 345 C
337 346 C
338 347 C
339 348 C
340 349 C
341 350 C
342 351 C
343 352 C
344 353 C
345 354 C
346 355 C
347 356 C
348 357 C
349 358 C
350 359 C
351 360 C
352 361 C
353 362 C
354 363 C
355 364 C
356 365 C
357 366 C
358 367 C
359 368 C
360 369 C
361 370 C
362 371 C
363 372 C
364 373 C
365 374 C
366 375 C
367 376 C
368 377 C
369 378 C
370 379 C
371 380 C
372 381 C
373 382 C
374 383 C
375 384 C
376 385 C
377 386 C
378 387 C
379 388 C
380 389 C
381 390 C
382 391 C
383 392 C
384 393 C
385 394 C
386 395 C
387 396 C
388 397 C
389 398 C
390 399 C
391 400 C
392 401 C
393 402 C
394 403 C
395 404 C
396 405 C
397 406 C
398 407 C
399 408 C
400 409 C
401 410 C
402 411 C
403 412 C
404 413 C
405 414 C
406 415 C
407 416 C
408 417 C
409 418 C
410 419 C
411 420 C
412 421 C
413 422 C
414 423 C
415 424 C
416 425 C
417 426 C
418 427 C
419 428 C
420 429 C
421 430 C
422 431 C
423 432 C
424 433 C
425 434 C
426 435 C
427 436 C
428 437 C
429 438 C
430 439 C
431 440 C
432 441 C
433 442 C
434 443 C
435 444 C
436 445 C
437 446 C
438 447 C
439 448 C
440 4
```

```

31 IF (ABS(I)-EPS*ABS(Q2)) 32,22,33
32 LL = 1
   GC = 12
   TEST ON DECREASING RELATIVE ERRORS
33 IF (L-1) 12,12,34
34 IF (ABS(IH)-EPS*I*ABS(Q1)) 35,35,12
35 IF (ABS(IH)-EPS*I*ABS(Q2)) 36,36,12
36 IF (ABS(CQCL*HH)-ABS(C1*DQ1)) 37,44,44
37 IF (ABS(CQCL*HH)-ABS(C2*DQ2)) 12,44,44
38 IF (ABS(CQ2*H)-ABS(C2*DQ2)) 12,44,44
39 END OF BAIRSTON ITERATION
   EXIT IN CASE OF QUADRATIC POLYNOMIAL
38 Q(1) = C(1)
   Q(2) = C(2)
   Q(3) = C(3)
   Q(4) = C(4)
   RETURN
   EXIT IN CASE OF SUFFICIENT ACCURACY
39 G(1) = C1
   G(2) = C2
   G(3) = A
   G(4) = B
   RETURN
   ERROR EXIT IN CASE OF ZERO OR CONSTANT POLYNOMIAL
40 IER = -1
   RETURN
   ERROR EXIT IN CASE OF LINEAR POLYNOMIAL
41 IER = -2
   RETURN
   ERROR EXIT IN CASE OF NONREFINED QUADRATIC FACTOR
42 IER = -3
   GC = 7C 44
   ERROR EXIT IN CASE OF UNSATISFACTORY ACCURACY
43 IER = 1
   G(1) = CQ1
   G(2) = CQ2
   G(3) = AA
   G(4) = BB
   RETURN
44 END

```

```

C+++++SUBROUTINE CCNORM
C+++++
C+++++PURPOSE: TO NORMALIZE THE COEFFICIENT FO THE HIGHEST POWER
C+++++TERM IN A POLYNOMIAL TO UNITY.
C+++++
C+++++CALLING SEQUENCE:
C+++++CALL XCNORM(V,N,XNCRM)
C+++++
C+++++DESCRIPTION OF PARAMETERS:
C+++++
C+++++V = ONE DIMENSIONAL ARRAY CONTAINING THE UNNORMALIZED
C+++++COEFFICIENTS IN ASCENDING POWER OF THE INDEPENDENT
C+++++VARIABLE. UPON RETURN FROM THE SUBROUTINE THIS
C+++++VECTOR WILL CONTAIN THE NORMALIZED COEFFICIENTS.
C+++++
C+++++N = DIMENSION OF THE ARRAY V.
C+++++
C+++++XNCRM = NORMALIZING FACTOR BY WHICH ALL COEFFICIENTS HAVE
C+++++BEEN DIVIDED.
C+++++
C+++++SLEROUTINES AND FUNCTION SUBPROGRAMS REQUIRED:
C+++++NONE
C+++++
C+++++SLEROUTINE CCNORM (V,N,VNORM)
C+++++DIMENSION V(1)
C+++++IF (VN) 1,3,1
C+++++
CINSURE HIGHEST ORDER COEFFICIENT IS NOT UNITY
C1 IF (V(N).EQ.1.0) GO TO 4
C
CSELECT COEFFICIENT OF HIGHEST ORDER TERM AS NORMALIZING FACTOR
CAND DIVIDE EACH COEFFICIENT OF POLYNOMIAL BY THIS FACTOR.
C
C VNCRM = V(N)
C
C2 I=1,N
C V(I) = V(I)/VNCRM
C2 CCNTINUE
C
CWRITE (6,5)
C3 RETURN
C4 VNCRM = 1.0
C4 RETURN
C

```

```

5  FCRMAT ('0',72('*'),//,' WARNING-----COEFFICIENT OF HIGHEST ORDER ICOND 450
1ERM IS ZERO',//,' EXECUTION OF SUBROUTINE CONCRM TERMINATING WITH HCCOND 500
2LT NORMALIZING',//,' THE COEFFICIENT VECTOR.',//,72('*')) COND 510
ENC COND 520

```



[illegible]

[illegible]



```

C
ITE(1) = 2
ITE(2) = 4
CALL CRAMP (NOMEG, FREQ, ADMAG, ITB, RTB)
ITE(1) = 2
ITE(2) = 1
CALL CRAMP (NOMEG, FREQ, AMAG, ITB, RTB)
ITE(1) = 3
ITE(2) = 0
CALL CRAMP (500, WFREQS, SAMAG, ITB, RTB)

C 18 I=1,12
   RTE(I+16) = TITLE2(I)
   18 CONTINUE
C
ITE(1) = 1
ITE(2) = 2
ITE(3) = 9
ITE(4) = 6
CALL CRAMP (2, SCALE2, SCALE1, ITB, RTB)
ITE(1) = 2
ITE(2) = 4
CALL CRAMP (NOMEG, FREQ, DQPHSD, ITB, RTB)
ITE(1) = 1
ITE(2) = 2
CALL CRAMP (NOMEG, FREQ, QPTSD, ITB, RTB)
ITE(1) = 3
ITE(2) = 0
CALL CRAMP (500, WFREQS, SAPHSD, ITB, RTB)
WRITE (6,35)
WRITE (6,45)
WRITE (6,35)
GC TO 21
15 WRITE (6,35)
WRITE (6,35)
WRITE (6,35)
GC TO 21
20 WRITE (6,35)
WRITE (6,40)
WRITE (6,35)
21 IF (NICHOL) 25,22,24
C 22 I=1,12
   RTE(I+16) = TITLE3(I)
   22 CONTINUE
C
ITE(1) = 1
ITE(2) = 2

```

```

GRAP1450
GRAP1460
GRAP1470
GRAP1480
GRAP1490
GRAP1500
GRAP1510
GRAP1520
GRAP1530
GRAP1540
GRAP1550
GRAP1560
GRAP1570
GRAP1580
GRAP1590
GRAP1600
GRAP1610
GRAP1620
GRAP1630
GRAP1640
GRAP1650
GRAP1660
GRAP1670
GRAP1680
GRAP1690
GRAP1700
GRAP1710
GRAP1720
GRAP1730
GRAP1740
GRAP1750
GRAP1760
GRAP1770
GRAP1780
GRAP1790
GRAP1800
GRAP1810
GRAP1820
GRAP1830
GRAP1840
GRAP1850
GRAP1860
GRAP1870
GRAP1880
GRAP1890
GRAP1900
GRAP1910
GRAP1920

```

```

C
24 WRITE(6,35)
    WRITE(6,42)
    WRITE(6,35)
25 WRITE(6,35)
    WRITE(6,43)
    WRITE(6,35)
26 IF (NYCST) 30,27,25
C
27 CC 28 I=1,12
    RTE(I+10) = TITLE4(I)
28 CCNTINUE
    ITE(1) = 0
    ITE(2) = 8
    ITE(3) = 8
    ITE(4) = 8
    CALL CRAMP(5)
    WRITE(6,35)
    WRITE(6,35)
    WRITE(6,35)
29 WCCITE(6,35)
    WCCITE(6,35)
    WCCITE(6,35)
30 WCCITE(6,35)
    WCCITE(6,35)
    WCCITE(6,35)
31 CCNTINUE
    RTE(I+10) = TITLE5(I)
C

```

[illegible]

```

FUNCTION KE(X)
DIMENSION X(24), Y(12), Z(12)
COMMON ACN, NSN, NSD, COFASN(40), COFASDI(40), CMAG(500), DOPHSD(500), M
1CFRFSR(500), GPHSD(500), N1,N2,N3,N4,P1,T,WS,I2DT,ACMAG(500),NTITLE,I
2E), FREQ(500), AMAG(500), NMINES, CCFMRK(40), IPLCT, WFRECS(500), SAPHSC
3C), SAMAG(500), DIFF1(500), DIFF2(500), ICCST, CMRK1(40)
4KE = 0
CC 1 I=1,N1
Y(I) = X(I)
1 CC CONTINUE
CC
CC 2 J=1,N2
Z(J) = X(N1+J)
2 CC CONTINUE
CC
CC COMPILE THE SYSTEM OPEN LCOP NUMERATOR AND DENOMINATOR POLYNOMIALS
CALL PMPY (COFASN,N7,Y,N1,A,N3)
CALL PMPY (COFASD,N8,Z,N2,B,N4)
NSN = N7-1
NSD = N8-1
CALL PVALL (COFASN,NSN,0.0,WFREQ(1),CNUMR,CNUMI)
CALL PVALL (COFASD,NSD,0.0,WFREQ(1),CDENR,CDEAI)
IF (ABS(CNUMR).GT.2.9E+37) K=1
IF (ABS(CNUMI).GT.2.9E+37) K=1
IF (ABS(CDENR).GT.2.9E+37) K=1
IF (ABS(CDEAI).GT.2.9E+37) K=1
IF (KE.EQ.1) WRITE(6,5)
IF (NMINES.EQ.1) GO TO 4
IF (CMRK1.EQ.1) GO TO 3
CC CHECK FOR RIGHT HALF PLANE ROOTS OF COMPENSATOR NUMERATOR
CC
      CALL PINVRT(Y,N1)
      CALL ROUTH(Y,N1,ISTN)
      CALL PINVRT(Y,N1)
      KE = ISTN
      IF (KE.EQ.1) GO TO 4
CC CHECK FOR RIGHT HALF PLANE ROOTS OF COMPENSATOR DENOMINATOR
CC
      CALL PINVRT(Z,N2)
      CALL ROUTH(Z,N2,ISTD)
      CALL PINVRT(Z,N2)
      KE = ISTD

```

```

4 IF (KE.EQ.1) GO TO 4
  RETURN
5 FCMPAT ('O',80,'-'),/, ,4(' * * * WARNING * * *'),/,O('THE SIZE
  IE CF THE REAL AND IMAGINARY PARTS OF THE NUMERATOR AND DENOMINATOR
  3 IE CF ARE CLOSE TO THE MAXIMUM VALUE THAT THE COMPUTER CAN HANDLE
  3 FREQUENCY/,M.,/,O,4(' * * * WARNING * * *'),/, ,80(' - , KE
  4 ELIMINATE THE PROBLEM.,/,O,4(' * * * WARNING * * *'),/, ,80(' - , KE
  5)
  END

```

5



```

400  GC TO 5
401  C MAG(K) = SCRT(QNUM1**2+QNUMR**2)/SQRT(QCEN1**2+QCENR**2)
402  C ACJUST MAGITUDE RESPONSE IF ZERO ORDER HOLD IS PRESENT IN THE SYSTEM
403  C
404  IF (IZJF.EQ.1) QMAG(K)=QMAG(K)*AES((SIN(PI*WREQ(K)/WS1))/(PI*WREQ
405  1(K)/WS1))
406  CR(K) = (CNUMR*QDENR+CNUM1*QCEN1)/(QCENR**2+QDEN1**2)
407  Q1(K) = (QNUM1*QDENR-CNUMR*QCEN1)/(QCENR**2+QDEN1**2)
408  5 IF ((CNUM EQ.0.0).AND.(CNUMR.EQ.0.0)) QNUMR=1.0
409  IF (CNUMR = ATAN2(QNUM1,CNUMR)
410  ANGNUR = ANGNUR*57.29578
411  IF (ANGAUD.LT.0.0) ANGNUD=360.0+ANGNUR
412  CELPHI = ANGNUD-PHI
413  IF (ICELNU.EQ.0) GO TO 8
414  IF ((DELPHI).GE.0.0.AND.(ICELNU.EQ.1)) GO TO 8
415  IF ((DELPHI).LT.0.0.AND.(ICELNU.EQ.-1)) GO TO 6
416  IF ((DELPHI).GE.0.0.AND.(ICELNU.EQ.-1)) GO TO 7
417  IF ((DELPHI).LT.0.0.AND.(ICELNU.EQ.1)) GO TO 8
418  IF ((ABS(DELPHI)).LT.180.0) GC TO 8
419  6 CELPHI = ANGNUD+360.0-PHI
420  GC TC 8
421  7 IF (ABS(DELPHI).LT.180.0) GO TO 8
422  CELPHI = ANGNUD-360.0-PHI
423  8 P1PHI = P1CHK+DELPHI
424  9 P1PHI = P1CHK/360.0
425  P1PHI = ANGNUD
426  ANGNUD=IPHI*360.0
427  ANGNUR = ATAN2(QDEN1,CDENR)
428  ANGCEC = ANGNUR*57.29578
429  IF (ANGCEC.LT.0.0) ANGCEC=360.0+ANGCEC
430  CELU (ICELNU.EQ.0) GC TC 11
431  IF ((DELPSI).GE.0.0.AND.(ICELNU.EQ.1)) GO TO 11
432  IF ((DELPSI).LT.0.0.AND.(ICELNU.EQ.-1)) GO TO 5
433  IF ((DELPSI).GE.0.0.AND.(ICELNU.EQ.-1)) GO TC 19
434  IF ((DELPSI).LT.0.0.AND.(ICELNU.EQ.1)) GO TC 11
435  5 IF (ABS(DELPSI).LT.180.0) GC TO 11
436  CELU (ICELNU.EQ.0) GC TC 11
437  IF ((DELPSI).LT.180.0) GC TO 11
438  CELU (ICELNU.EQ.1) GC TO 11
439  IF ((DELPSI).LT.180.0) GC TO 11
440  CELU (ICELNU.EQ.-1) GC TC 19
441  10 IF (ABS(DELPSI).LT.180.0) GC TO 11
442  CELU (ICELNU.EQ.0) GC TC 11
443  11 P1PHI = P1CHK+DELPSI
444  P1PHI = P1CHK/360.0
445  IF (P1PHI = ANGCEC)
446  P1PHI = ANGCEC+IPHI*360.0
447  IF ((DELPHI).GE.0.0) IDELNU=1
448  IF ((DELPHI).LT.0.0) IDELNU=-1

```

```

570 IF (DELPS1-GE.0.0) ICELDE=1
580 IF (DELPS1-LT.0.0) IDELDE=-1
590 CPHSD(K) = ANGNUD-ANGDEC
1000 CPHSR(K) = ANGNUR-ANGDER
1010 IF (I2OLF-NE.1) GO TO 12
1020 IWS = W*FREQ(K)/WS
1030
1040
1050
1060
1070
1080
1090
1100
1110
1120
1130
1140
1150
1160
1170
1180
1190
1200
1210
1220
1230
1240
1250

```

C ACJLST PHASE RESPONSE IF ZERO ORDER HCLD IS PRESENT IN THE SYSTEM  
C

```

12 CPHSC(K) = QPHSD(K) - (FI*W*FREQ(K)/WS)*57.2557E-IWS*180.0
13 IF (ICOST) 14,14,13
14 GO TO (14,15,16,17,18,19), ICOST
15 CCST = COST + (1.0-QMAG(K))/DMAG(K)**2 + (1.0-QPHSD(K)/DCPHSD(K))**2
16 GC TC 20
17 CCST = COST + (1.0-QMAG(K))/DMAG(K)**2
18 GC TO 20
19 CCST = COST + (1.0-QPHSD(K))/DCPHSD(K)**2
20 GC TO 20
21 CCST = COST + ABS(1.0-QMAG(K))/DMAG(K)) + ABS(1.0-C*CPHSD(K)/DCPHSD(K))
22 GC TO 20
23 CCST = COST + ABS(1.0-QMAG(K))/DMAG(K))
24 GC TO 20
25 CCST = COST + ABS(1.0-QPHSD(K))/DQPHSD(K))
26 CCNTINUE

```

C  
C

```

FE = COST
RETURN
END

```



```

C+++++SUBROUTINE PMPY
C+++++
C+++++PURPOSE: TO MULTIPLY THE COEFFICIENTS OF TWO POLYNOMIALS AND
C+++++RETURN THE COEFFICIENTS OF THE RESULTING POLYNOMIAL.
C+++++POLYNOMIALS TO BE MULTIPLIED MUST HAVE REAL
C+++++COEFFICIENTS, BUT NEED NOT BE OF THE SAME ORDER.
C+++++
C+++++CALLING SEQUENCE:
C+++++CALL PMPY(Z, IDIMZ, X, IDIMX, Y, IDIMY)
C+++++
C+++++DESCRIPTION OF PARAMETERS:
C+++++Z = VECTOR OF RESULTANT COEFFICIENTS, ORDERED IN
C+++++ASCENDING POWERS OF THE INDEPENDENT VARIABLE.
C+++++IDIMZ = CALCULATED DIMENSION OF THE RESULTING POLYNOMIAL
C+++++VECTOR.
C+++++X = VECTOR OF COEFFICIENTS OF FIRST POLYNOMIAL TO BE
C+++++MULTIPLIED IN ASCENDING POWERS OF THE INDEPENDENT
C+++++VARIABLE.
C+++++IDIMX = DIMENSION OF THE VECTOR X.
C+++++Y = VECTOR OF COEFFICIENTS OF THE SECOND POLYNOMIAL TO
C+++++BE MULTIPLIED ORDERED IN ASCENDING POWERS OF THE
C+++++INDEPENDENT VARIABLE.
C+++++IDIMY = DIMENSION OF THE VECTOR Y.
C+++++
C+++++SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED:
C+++++NCNE
C+++++
C+++++METHOD:
C+++++THE COEFFICIENTS OF THE RESULTANT POLYNOMIAL ARE CALCULATED
C+++++AS THE SUM OF PRODUCTS OF THE COEFFICIENTS OF THE INPUT
C+++++POLYNOMIALS WHOSE EXPONENTS ADD UP TO FORM THE CORRESPONDING
C+++++EXPONENT OF THE RESULTING COEFFICIENTS.
C+++++
C+++++SUBROUTINE PMPY (Z, IDIMZ, X, IDIMX, Y, IDIMY)
C+++++DIMENSION Z(1), X(1), Y(1)
C+++++
C+++++IF (IDIMX*IDIMY) 1,1,2
C+++++1 ICIMZ = 0
C+++++GO TO 5
C+++++2 ICIMZ = IDIMX+IDIMY-1
C+++++
C+++++CC 3 I=1, ICIMZ
C+++++3 Z(I) = 0.

```

```

C
C      CC 4 I=1, IDIMX
C      CC 4 J=1, IDIMY
C      K = I+J-1
C      4 Z(K) = X(I)*Y(J)+Z(K)
C      5 RETURN
C      6 ENCL

```

```

4900
5000
5100
5200
5300
5400
5500
5600
5700
5800

```



AND NUMBER OF IMPLICIT CONSTRAINT EVALUATIONS ARE INCLUDED IN THE OUTPUT.  
ADDITIONALLY, (WHEN NPR .GT. 0) THE SAME INFORMATION WILL BE OUTPUT:

- 1) IF THE INITIAL POINT IS NOT FEASIBLE
- 2) AFTER THE FIRST COMPLETE COMPLEXITY GENERATED TRIAL,
- 3) IF A FEASIBLE VERTEX CANNOT BE FOUND AT SOME TRIAL,
- 4) IF THE OBJECTIVE VALUE OF A VERTEX CANNOT BE MALE
- 5) NO-LOCAL-MINIMUM TRIALS (NTA) IS REACHED AND, CEC FOR
- 6) WHEN THE LIMIT ON TRIALS HAS BEEN UNCHANGED FOR
- 2\*NV TRIALS, INDICATING A LOCAL MINIMUM HAS BEEN FOUND.

IF THE USER WISHES TO TRACE THE PROGRESS OF A SOLUTION, A CHOICE OF NPR = 25, 50 OR 100 IS RECOMMENDED.

NTA INTEGER INPUT OF LIMIT ON THE NUMBER OF TRIALS ALLOWED IN THE CALCULATION. IF THE USER INPUTS NTA.LE.0, A DEFAULT VALUE OF 2000 IS USED. WHEN THIS LIMIT IS REACHED, CONTROL RETURNS TO THE CALLING PROGRAM WITH THE BEST ATTAINED OBJECTIVE FUNCTION VALUE IN YMN, AND THE BEST ATTAINED SOLUTION POINT IN XS.

R A REAL NUMBER INPUT TO DEFINE THE FIRST RANDCM NUMBER USED IN DEVELOPING THE INITIAL COMPLEX OF 2\*NV VERTICES. (0., .GT. R.LT. 1.) IF R IS NOT WITHIN THESE BOUNDS, IT WILL BE REPLACED BY 1./3..

XS INPUT REAL ARRAY DIMENSIONED AT LEAST NV+NAV. THE FIRST NV MUST CONTAIN A FEASIBLE ORIGIN FOR STARTING THE CALCULATION. THE LAST NAV NEED NOT BE INITIALIZED. THE CALL-RETURN FROM ECXPLX, THE FIRST NV ELEMENTS OF THE UPPER CONTAIN THE COORDINATES OF THE MINIMUM OBJECTIVE FUNCTION, AND THE REMAINING NAV (NAV.GE. 0) CONTAIN THE VALUES OF THE CORRESPONDING AUXILIARY VARIABLES.

IP INTEGER INPUT FOR OPTIONAL INTEGER PROGRAMMING. IF IP=1, THE VALUES OF THE INDEPENDENT VARIABLES WILL BE REPLACED WITH INTEGER VALUES (STILL STORED AS REAL\*4).

XU A REAL ARRAY DIMENSIONED AT LEAST NV INPUTTING THE UPPER BOUND ON EACH INDEPENDENT VARIABLE, (EACH EXPLICIT CCN- STRAINT). INPUT VALUES ARE SLIGHTLY ALTERED BY ECXPLX.

XL A REAL ARRAY DIMENSIONED AT LEAST NV INPUTTING THE LOWER BOUND ON EACH INDEPENDENT VARIABLE, (EACH EXPLICIT CCN-

490 BXFL  
500 BXPL  
510 BXPL  
520 BXPL  
530 BXPL  
540 BXPL  
550 BXPL  
560 BXPL  
570 BXPL  
580 BXFL  
590 BXFL  
600 BXFL  
610 BXFL  
620 BXFL  
630 BXFL  
640 BXFL  
650 BXFL  
660 BXPL  
670 BXPL  
680 BXPL  
690 BXPL  
700 BXPL  
710 BXPL  
720 BXPL  
730 BXPL  
740 BXPL  
750 BXPL  
760 BXPL  
770 BXPL  
780 BXPL  
790 BXPL  
800 BXPL  
810 BXPL  
820 BXPL  
830 BXPL  
840 BXPL  
850 BXPL  
860 BXPL  
870 BXPL  
880 BXPL  
890 BXPL  
900 BXPL  
910 BXPL  
920 BXPL  
930 BXPL  
940 BXPL  
950 BXPL  
960 BXPL

BXPL 970  
 BXPL 980  
 BXPL 990  
 BXPL 1000  
 BXPL 1010  
 BXPL 1020  
 BXPL 1030  
 BXPL 1040  
 BXPL 1050  
 BXPL 1060  
 BXPL 1070  
 BXPL 1080  
 BXPL 1090  
 BXPL 1100  
 BXPL 1110  
 BXPL 1120  
 BXPL 1130  
 BXPL 1140  
 BXPL 1150  
 BXPL 1160  
 BXPL 1170  
 BXPL 1180  
 BXPL 1190  
 BXPL 1200  
 BXPL 1210  
 BXPL 1220  
 BXPL 1230  
 BXPL 1240  
 BXPL 1250  
 BXPL 1260  
 BXPL 1270  
 BXPL 1280  
 BXPL 1290  
 BXPL 1300  
 BXPL 1310  
 BXPL 1320  
 BXPL 1330  
 BXPL 1340  
 BXPL 1350  
 BXPL 1360  
 BXPL 1370  
 BXPL 1380  
 BXPL 1390  
 BXPL 1400  
 BXPL 1410  
 BXPL 1420  
 BXPL 1430  
 BXPL 1440

STRAINT). NOTE: FOR BCTP XU AND XL CHOOSE REASCALABLE  
 VALUES IF NONE ARE GIVEN, NOT VALUES WHICH ARE MAGNITUDES  
 ABOVE OR BELOW THE EXPECTED SOLUTION. INPUT VALUES ARE  
 SLIGHTLY ALTERED BY BOXPLX.

YMN THIS OUTPUT IS THE VALUE (REAL\*4) OF THE OBJECTIVE FUNC-  
 TION, CORRESPONDING TO THE SOLUTION PCINT OUTPUT IN XS.  
 IER INTEGER ERROR RETURN. TO BE INTERRUPTED UCN RETURN  
 FROM BOXPLX. IER WILL BE ONE OF THE FOLLOWING:

=-1 CANNOT FIND FEASIBLE VERTEX OR FEASIBLE CENTROID  
 AT THE START OR A RESTART (SEE 'METHODS' BELOW)  
 =0 FUNCTION VALUE UNCHANGED FOR 'N' TRIALS (HERE  
 N=6\*NV+10) THIS IS THE NORMAL RETURN PARAMETER.  
 =1 CANNOT DEVELOP FEASIBLE VERTEX.  
 =2 CANNOT DEVELOP A NO-LONGER-WORKING VERTEX.  
 =3 LIMIT ON TRIALS REACHED. (NTA EXCEEDED)  
 NOTE: VALID RESULTS MAY BE RETURNED IN ANY OF THE  
 ABOVE CASES.

# EXAMPLE OF USAGE

THIS EXAMPLE MINIMIZES THE OBJECTIVE FUNCTION SPCHN IN THE  
 EXTERNAL FUNCTION FE(X). THERE ARE TWO INDEPENDENT VAR-  
 IABLES X(1) & X(2), AND TWO IMPLICIT CONSTRAINT FUNCTIONS  
 X(3) & X(4) WHICH ARE EVALUATED AS AUXILIARY VARIABLES (SEE  
 EXTERNAL FUNCTION KE(X)).

DIMENSION XS(4),XU(2),XL(2)

STARTING GUESS

XS(1) = 1.0

XS(2) = 0.5

UPPER LIMITS

XL(1) = 6.0

XL(2) = 6.0

LOWER LIMITS

XL(1) = 0.0

XL(2) = 0.0

R = 9./13.

NFR = 500

NAV = 2

IF = 0







VALUES DECLARED BY THE USER. THIS ADJUSTMENT IS NOT MADE WHEN IP=1.

NOTE: NO NON-LINEAR PROGRAMMING ALGORITHM CAN GUARANTEE THAT THE ANSWER FOUND IS THE GLOBAL MINIMUM, RATHER THAN JUST A LOCAL MINIMUM. HOWEVER, ACCORDING TO REF. 2, THE COMPLEX METHOD HAS AN ADVANTAGE IN THAT IT TENDS TO FIND THE GLOBAL MINIMUM MORE FREQUENTLY THAN MANY OTHER NON-LINEAR PROGRAMMING ALGORITHMS.

IT SHOULD BE NOTED THAT THE AUXILIARY VARIABLE FEATURE CAN ALSO BE USED TO DEAL WITH PROBLEMS CONTAINING EQUALITY CONSTRAINTS. ANY EQUALITY CONSTRAINT IMPLIES THAT A GIVEN VARIABLE IS NOT TRULY INDEPENDENT. THEREFORE, IN GENERAL, ONE VARIABLE INVOLVED IN AN EQUALITY CONSTRAINT CAN BE REMOVED FROM THE SET OF NV INDEPENDENT VARIABLES AND ADDED TO THE SET OF NV AUXILIARY VARIABLES. THIS USUALLY INVOLVES RENUMBERING THE INDEPENDENT VARIABLES OF THE GIVEN PROBLEM.

# SUBROUTINES AND FUNCTIONS REQUIRED

SUBROUTINE 'BOUN' AND FUNCTION 'FBV' ARE INTEGRAL PARTS OF THE EOXPX PACKAGE.

TWO FUNCTIONS MUST BE SUPPLIED BY THE USER. THE FIRST, KE(X), IS USED TO EVALUATE THE IMPLICIT CONSTRAINTS SET AT THE BEGINNING OF THE FUNCTION. THEN EVALUATE THE IMPLICIT CONSTRAINTS WITHIN THE RANGE (0.0, 1.0). THE SECOND, CONSTRAINT X(4), MUST BE 0.0 IF EITHER CONSTRAINT IS NOT WITHIN THESE BOUNDS. CONTROL IS TRANSFERRED TO STATEMENT 1, AND KE IS SET TO "1" AND CONTROL IS RETURNED TO EOXPX.

THE SECOND FUNCTION THE USER MUST PROVIDE EVALUATES THE OBJECTIVE FUNCTION. IT IS CALLED F(X) AS SHOWN IN THE EXAMPLE ABOVE, AND KE MUST BE SET TO THE VALUE OF THE OBJECTIVE FUNCTION CORRESPONDING TO CURRENT VALUES OF THE NV INDEPENDENT VARIABLES IN ARRAY 'X'.

## REFERENCES

- FOX, M. J., "A NEW METHOD OF CONSTRAINED OPTIMIZATION AND A COMPARISON WITH OTHER METHODS", COMPUTER JOURNAL, 8 APR. '65, PP. 45-52.
- BEVERIDGE G. J. AND SCHECHTER R., "OPTIMIZATION: THEORY AND PRACTICE", MCGRAW-HILL, 1970.





```

BXPL3370
BXPL3380
BXPL3390
BXPL3400
BXPL3410
BXPL3420
BXPL3430
BXPL3440
BXPL3450
BXPL3460
BXPL3470
BXPL3480
BXPL3490
BXPL3500
BXPL3510
BXPL3520
BXPL3530
BXPL3540
BXPL3550
BXPL3560
BXPL3570
BXPL3580
BXPL3590
BXPL3600
BXPL3610
BXPL3620
BXPL3630
BXPL3640
BXPL3650
BXPL3660
BXPL3670
BXPL3680
BXPL3690
BXPL3700
BXPL3710
BXPL3720
BXPL3730
BXPL3740
BXPL3750
BXPL3760
BXPL3770
BXPL3780
BXPL3790
BXPL3800
BXPL3810
BXPL3820
BXPL3830
BXPL3840

```

```

3 IF (NPR.GT.0) WRITE (6,51) II
4 CEN(I) = VT
5 IF (IP.EQ.1) GO TO 4
6 BL(I) = BL(I)+MAX1(EP,EP*ABS(BL(I)))
7 BL(I) = BU(I)-AMAX1(EP,EP*ABS(BU(I)))
8 SLM(I) = VT
9
10 C
11 C
12 C
13 C
14 C
15 C
16 C
17 C
18 C
19 C
20 C
21 C
22 C
23 C
24 C
25 C
26 C
27 C
28 C
29 C
30 C
31 C
32 C
33 C
34 C
35 C
36 C
37 C
38 C
39 C
40 C
41 C
42 C
43 C
44 C
45 C
46 C
47 C
48 C
49 C
50 C
51 C
52 C
53 C
54 C
55 C
56 C
57 C
58 C
59 C
60 C
61 C
62 C
63 C
64 C
65 C
66 C
67 C
68 C
69 C
70 C
71 C
72 C
73 C
74 C
75 C
76 C
77 C
78 C
79 C
80 C
81 C
82 C
83 C
84 C
85 C
86 C
87 C
88 C
89 C
90 C
91 C
92 C
93 C
94 C
95 C
96 C
97 C
98 C
99 C
100 C
101 C
102 C
103 C
104 C
105 C
106 C
107 C
108 C
109 C
110 C
111 C
112 C
113 C
114 C
115 C
116 C
117 C
118 C
119 C
120 C
121 C
122 C
123 C
124 C
125 C
126 C
127 C
128 C
129 C
130 C
131 C
132 C
133 C
134 C
135 C
136 C
137 C
138 C
139 C
140 C
141 C
142 C
143 C
144 C
145 C
146 C
147 C
148 C
149 C
150 C
151 C
152 C
153 C
154 C
155 C
156 C
157 C
158 C
159 C
160 C
161 C
162 C
163 C
164 C
165 C
166 C
167 C
168 C
169 C
170 C
171 C
172 C
173 C
174 C
175 C
176 C
177 C
178 C
179 C
180 C
181 C
182 C
183 C
184 C
185 C
186 C
187 C
188 C
189 C
190 C
191 C
192 C
193 C
194 C
195 C
196 C
197 C
198 C
199 C
200 C
201 C
202 C
203 C
204 C
205 C
206 C
207 C
208 C
209 C
210 C
211 C
212 C
213 C
214 C
215 C
216 C
217 C
218 C
219 C
220 C
221 C
222 C
223 C
224 C
225 C
226 C
227 C
228 C
229 C
230 C
231 C
232 C
233 C
234 C
235 C
236 C
237 C
238 C
239 C
240 C
241 C
242 C
243 C
244 C
245 C
246 C
247 C
248 C
249 C
250 C
251 C
252 C
253 C
254 C
255 C
256 C
257 C
258 C
259 C
260 C
261 C
262 C
263 C
264 C
265 C
266 C
267 C
268 C
269 C
270 C
271 C
272 C
273 C
274 C
275 C
276 C
277 C
278 C
279 C
280 C
281 C
282 C
283 C
284 C
285 C
286 C
287 C
288 C
289 C
290 C
291 C
292 C
293 C
294 C
295 C
296 C
297 C
298 C
299 C
300 C
301 C
302 C
303 C
304 C
305 C
306 C
307 C
308 C
309 C
310 C
311 C
312 C
313 C
314 C
315 C
316 C
317 C
318 C
319 C
320 C
321 C
322 C
323 C
324 C
325 C
326 C
327 C
328 C
329 C
330 C
331 C
332 C
333 C
334 C
335 C
336 C
337 C
338 C
339 C
340 C
341 C
342 C
343 C
344 C
345 C
346 C
347 C
348 C
349 C
350 C
351 C
352 C
353 C
354 C
355 C
356 C
357 C
358 C
359 C
360 C
361 C
362 C
363 C
364 C
365 C
366 C
367 C
368 C
369 C
370 C
371 C
372 C
373 C
374 C
375 C
376 C
377 C
378 C
379 C
380 C
381 C
382 C
383 C
384 C
385 C
386 C
387 C
388 C
389 C
390 C
391 C
392 C
393 C
394 C
395 C
396 C
397 C
398 C
399 C
400 C
401 C
402 C
403 C
404 C
405 C
406 C
407 C
408 C
409 C
410 C
411 C
412 C
413 C
414 C
415 C
416 C
417 C
418 C
419 C
420 C
421 C
422 C
423 C
424 C
425 C
426 C
427 C
428 C
429 C
430 C
431 C
432 C
433 C
434 C
435 C
436 C
437 C
438 C
439 C
440 C
441 C
442 C
443 C
444 C
445 C
446 C
447 C
448 C
449 C
450 C
451 C
452 C
453 C
454 C
455 C
456 C
457 C
458 C
459 C
460 C
461 C
462 C
463 C
464 C
465 C
466 C
467 C
468 C
469 C
470 C
471 C
472 C
473 C
474 C
475 C
476 C
477 C
478 C
479 C
480 C
481 C
482 C
483 C
484 C
485 C
486 C
487 C
488 C
489 C
490 C
491 C
492 C
493 C
494 C
495 C
496 C
497 C
498 C
499 C
500 C
501 C
502 C
503 C
504 C
505 C
506 C
507 C
508 C
509 C
510 C
511 C
512 C
513 C
514 C
515 C
516 C
517 C
518 C
519 C
520 C
521 C
522 C
523 C
524 C
525 C
526 C
527 C
528 C
529 C
530 C
531 C
532 C
533 C
534 C
535 C
536 C
537 C
538 C
539 C
540 C
541 C
542 C
543 C
544 C
545 C
546 C
547 C
548 C
549 C
550 C
551 C
552 C
553 C
554 C
555 C
556 C
557 C
558 C
559 C
560 C
561 C
562 C
563 C
564 C
565 C
566 C
567 C
568 C
569 C
570 C
571 C
572 C
573 C
574 C
575 C
576 C
577 C
578 C
579 C
580 C
581 C
582 C
583 C
584 C
585 C
586 C
587 C
588 C
589 C
590 C
591 C
592 C
593 C
594 C
595 C
596 C
597 C
598 C
599 C
600 C
601 C
602 C
603 C
604 C
605 C
606 C
607 C
608 C
609 C
610 C
611 C
612 C
613 C
614 C
615 C
616 C
617 C
618 C
619 C
620 C
621 C
622 C
623 C
624 C
625 C
626 C
627 C
628 C
629 C
630 C
631 C
632 C
633 C
634 C
635 C
636 C
637 C
638 C
639 C
640 C
641 C
642 C
643 C
644 C
645 C
646 C
647 C
648 C
649 C
650 C
651 C
652 C
653 C
654 C
655 C
656 C
657 C
658 C
659 C
660 C
661 C
662 C
663 C
664 C
665 C
666 C
667 C
668 C
669 C
670 C
671 C
672 C
673 C
674 C
675 C
676 C
677 C
678 C
679 C
680 C
681 C
682 C
683 C
684 C
685 C
686 C
687 C
688 C
689 C
690 C
691 C
692 C
693 C
694 C
695 C
696 C
697 C
698 C
699 C
700 C
701 C
702 C
703 C
704 C
705 C
706 C
707 C
708 C
709 C
710 C
711 C
712 C
713 C
714 C
715 C
716 C
717 C
718 C
719 C
720 C
721 C
722 C
723 C
724 C
725 C
726 C
727 C
728 C
729 C
730 C
731 C
732 C
733 C
734 C
735 C
736 C
737 C
738 C
739 C
740 C
741 C
742 C
743 C
744 C
745 C
746 C
747 C
748 C
749 C
750 C
751 C
752 C
753 C
754 C
755 C
756 C
757 C
758 C
759 C
760 C
761 C
762 C
763 C
764 C
765 C
766 C
767 C
768 C
769 C
770 C
771 C
772 C
773 C
774 C
775 C
776 C
777 C
778 C
779 C
780 C
781 C
782 C
783 C
784 C
785 C
786 C
787 C
788 C
789 C
790 C
791 C
792 C
793 C
794 C
795 C
796 C
797 C
798 C
799 C
800 C
801 C
802 C
803 C
804 C
805 C
806 C
807 C
808 C
809 C
810 C
811 C
812 C
813 C
814 C
815 C
816 C
817 C
818 C
819 C
820 C
821 C
822 C
823 C
824 C
825 C
826 C
827 C
828 C
829 C
830 C
831 C
832 C
833 C
834 C
835 C
836 C
837 C
838 C
839 C
840 C
841 C
842 C
843 C
844 C
845 C
846 C
847 C
848 C
849 C
850 C
851 C
852 C
853 C
854 C
855 C
856 C
857 C
858 C
859 C
860 C
861 C
862 C
863 C
864 C
865 C
866 C
867 C
868 C
869 C
870 C
871 C
872 C
873 C
874 C
875 C
876 C
877 C
878 C
879 C
880 C
881 C
882 C
883 C
884 C
885 C
886 C
887 C
888 C
889 C
890 C
891 C
892 C
893 C
894 C
895 C
896 C
897 C
898 C
899 C
900 C
901 C
902 C
903 C
904 C
905 C
906 C
907 C
908 C
909 C
910 C
911 C
912 C
913 C
914 C
915 C
916 C
917 C
918 C
919 C
920 C
921 C
922 C
923 C
924 C
925 C
926 C
927 C
928 C
929 C
930 C
931 C
932 C
933 C
934 C
935 C
936 C
937 C
938 C
939 C
940 C
941 C
942 C
943 C
944 C
945 C
946 C
947 C
948 C
949 C
950 C
951 C
952 C
953 C
954 C
955 C
956 C
957 C
958 C
959 C
960 C
961 C
962 C
963 C
964 C
965 C
966 C
967 C
968 C
969 C
970 C
971 C
972 C
973 C
974 C
975 C
976 C
977 C
978 C
979 C
980 C
981 C
982 C
983 C
984 C
985 C
986 C
987 C
988 C
989 C
990 C
991 C
992 C
993 C
994 C
995 C
996 C
997 C
998 C
999 C
1000 C

```

```

C C      ENCL CALCULATION IF FEASIBLE CENTROID CANNOT BE FOUND.
C C      IF (LIMIT.GE.NLIM) GO TO 11
C C
C C      CC E J=1,NV
C C
C C      RANCOM NUMBER GENERATOR (RANDU)
C C      ICR = ICR*65539
C C      IF (ICR.LT.O) IQR = IQR+2147482647+1
C C      RCX = ICR
C C      RCX = RCX*.4656613E-9
C C      V(J,I) = BL(J)+RCX*(BL(J)-BL(J))
C C      IF (IP.EQ.1) V(J,I)=AINT(V(J,I)+.5)
C C      E CCNTINCE
C C
C C      CC IO L=1,NLIM
C C      NCE = NCE+1
C C      IF (KE(V(1,I)).EQ.O) GO TO 12
C C
C C      CC S J=1,NV
C C      VT = .5*(V(J,I)+CEN(J))
C C      IF (IP.EQ.1) VT = AINT(VT+.5)
C C      V(J,I) = VT
C C      S CCNTINCE
C C
C C      IC CCNTINCE
C C
C C      11 IF (NPR.LE.O) GO TO 12
C C      WRITE (C53) I
C C      CALL BCCT (NT,NPT,NFE,NCE,NV,NVT,V,I,FUN,CEN,I)
C C      IER = -1
C C      GC TO 50
C C
C C      12 CC 14 J=1,NV
C C      SUM(J) = SUM(J)+V(J,I)
C C      14 CEN(J) = SUM(J)/FI
C C
C C      TRY TO ASSURE FEASIBLE CENTROID FOR STARTING.
C C      NCE = NCE+1
C C      IF (KE(CEN).EQ.O) GO TO 16

```

```

C      CC 15 J=1,NV
C      15 SUM(J) = SUM(J)-V(J,I)
C
C      CC 16 NFE = NFE+1
C      16 NFE = NFE+1
C      17 CCNTINUE
C
C      CC 18 I=2,K
C      18 IF (FUN(J).GE.FUN(I)) GO TO 18
C      18 J = I
C      18 CCNTINUE
C
C      CC 19 BASIC LOOP. ELIMINATE EACH WORST VERTEX IN TURN. IT MUST BECOME
C      19 NC (LARGER WORST, NOT MERELY IMPROVED. FIND NEXT-TO-WORST VERTEX,
C      19 THE 'JN'TH ONE.
C      19 JN = 1
C      19 IF (J.EC.1) JN = 2
C
C      CC 20 I=1,K
C      20 IF (I.EC.J) GO TO 20
C      20 IF (FUN(JN).GE.FUN(I)) GO TO 20
C      20 JN = I
C      20 CCNTINUE
C
C      CC 21 LIMIT = NUMBER OF MOVES DURING THIS TRIAL TOWARD THE CENTROID
C      21 CLC TC FUNCTION VALUE.
C      21 LIMIT = 1
C
C      CC 22 COMPLETE CENTROID AND EVER REFLECT WORST VERTEX.
C      22
C      CC 23 CC 21 I=1,NV
C      23 VT = V(I,J)
C      23 SUM(I) = SUM(I)-VT

```

```

BXPL4330
BXPL4340
BXPL4350
BXPL4360
BXPL4370
BXPL4380
BXPL4390
BXPL4400
BXPL4410
BXPL4420
BXPL4430
BXPL4440
BXPL4450
BXPL4460
BXPL4470
BXPL4480
BXPL4490
BXPL4500
BXPL4510
BXPL4520
BXPL4530
BXPL4540
BXPL4550
BXPL4560
BXPL4570
BXPL4580
BXPL4590
BXPL4600
BXPL4610
BXPL4620
BXPL4630
BXPL4640
BXPL4650
BXPL4660
BXPL4670
BXPL4680
BXPL4690
BXPL4700
BXPL4710
BXPL4720
BXPL4730
BXPL4740
BXPL4750
BXPL4760
BXPL4770
BXPL4780
BXPL4790
BXPL4800

```

```

BXPL4810
BXPL4820
BXPL4830
BXPL4840
BXPL4850
BXPL4860
BXPL4870
BXPL4880
BXPL4890
BXPL4900
BXPL4910
BXPL4920
BXPL4930
BXPL4940
BXPL4950
BXPL4960
BXPL4970
BXPL4980
BXPL4990
BXPL5000
BXPL5010
BXPL5020
BXPL5030
BXPL5040
BXPL5050
BXPL5060
BXPL5070
BXPL5080
BXPL5090
BXPL5100
BXPL5110
BXPL5120
BXPL5130
BXPL5140
BXPL5150
BXPL5160
BXPL5170
BXPL5180
BXPL5190
BXPL5200
BXPL5210
BXPL5220
BXPL5230
BXPL5240
BXPL5250
BXPL5260
BXPL5280

```

```

CEN(I) = SUM(I)/FKM
VT = BETA*CEN(I) - ALPHA*VT
IF (IP.EQ.1) VT = AINT(VT+.5)

INSURE THE EXPLICIT CONSTRAINTS ARE OBSERVED.
21 V(I,J) = ANAX1(AMIN1(VT,BU(I)),BL(I))

NT = NT+1

CHECK FOR IMPLICIT CONSTRAINT VIOLATION.

22 CC 27 N=1,NLIM
NCE = NCE+1
IF (KE(V(I,J)).EQ.0) GO TO 28

EVERY *KV*TH TIME, OVER-REFLECT THE OFFENDING VERTEX THROUGH THE
BEST VERTEX.
IF (MCC(N,KV).NE.0) GO TO 24
CALL FBV (K,FUN,M)

CC 23 I=1,NV
VT = BETA*V(I,M) - ALPHA*V(I,J)
IF (IP.EQ.1) VT = AINT(VT+.5)
23 V(I,J) = ANAX1(AMIN1(VT,BU(I)),BL(I))

GC TO 26

CONSTRAINT VIOLATION: MOVE NEW POINT TOWARD CENTROID.

24 CC 25 I=1,NV
VT = .5*(CEN(I)+V(I,J))
IF (IP.EQ.1) VT = AINT(VT+.5)
V(I,J) = VT
25 CCNTINUE

26 NT = NT+1
27 CCNTINUE

IEF = 1

CANACT GET FEASIBLE VERTEX BY MOVING TOWARD CENTROID,

```

```

C      CR BY OVER-REFLECTING THRU THE BEST VERTEX.
C      IF (NPR.LE.0) GO TO 44
C      WRITE (6,54) NT,J
C      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,J)
C      GO TO 44
C
C      FEASIBLE VERTEX FOUND, EVALUATE THE OBJECTIVE FUNCTION.
C      2E NFE=NFE+1
C      FUNTRY = FEIV(1,J)
C
C      TEST TO SEE IF FUNCTION VALUE HAS NOT CHANGED.
C      AFC = ABS(FUNTRY-FUNOLD)
C      AFX = AMAX1(ABS(EF*FUNCLD),EPI)
C
C      ACTIVATE THE FOLLOWING TWO STATEMENTS FOR DIAGNOSTIC PURPOSES ONLY.
C      55 FFORMAT (1X,I3,6E15.7,2I5)
C      IF (AFC.GT.AMX) GO TO 29
C      IF (NPTS=1)
C      IF (NTFS.LT.NCT) GO TO 30
C      ICR = 0
C      IF (NPR.LE.0) GO TO 44
C      WRITE (6,55) K
C      GO TO 44
C      29 NPTS = 0
C
C      IS THE NEW VERTEX NEAR LONGER WORST?
C      30 IF (FUNTRY.LT.FUN(JN)) GO TO 36
C
C      TRIAL VERTEX IS STILL WORST; ADJUST TOWARD CENTROID.
C      EVERY .KV. TH TIME, OVER-REFLECT THE OFFENDING VERTEX THROUGH THE
C      BEST VERTEX.
C      LIMIT=LIMIT+1
C      IF (NCO(LIMIT,KV).NE.0) GO TO 32
C      CALL FEV (K,FUN,M)
C
C      CC 31 I=1,NV
C      VT = BETA*V(I,M)-ALPHA*V(I,J)
C      IF (IP.EQ.1) VT = AINT(VT+.5)
C      31 V(I,J) = AMAX1(AMIN1(VT,BU(I)),BL(I))
C
C      GO TO 34
C
C      CC 32 I=1,NV
C      VT = .5*(CEN(I)+V(I,J))

```

```

XPL5770
XPL5780
XPL5790
XPL5800
XPL5810
XPL5820
XPL5830
XPL5840
XPL5850
XPL5860
XPL5870
XPL5880
XPL5890
XPL5900
XPL5910
XPL5920
XPL5930
XPL5940
XPL5950
XPL5960
XPL5970
XPL5980
XPL5990
XPL6000
XPL6010
XPL6020
XPL6030
XPL6040
XPL6050
XPL6060
XPL6070
XPL6080
XPL6090
XPL6100
XPL6110
XPL6120
XPL6130
XPL6140
XPL6150
XPL6160
XPL6170
XPL6180
XPL6190
XPL6200
XPL6210
XPL6220
XPL6230
XPL6240

```

```

      IF (IP.EQ.1) VT = AINT(VI+.5)
      V(I,J) = VI
22    CONTINUE

234   IF (LIMIT.LT.NLIM) GO TO 35
      (CANNOT MAKE THE 'J' TH VERTEX NO LONGER WORST BY DISPLACING TOWARD
       THE CENTROID OR BY COVER-REFLECTING THRU THE BEST VERTEX.)
      IFR = 2
      IF (NPR.GT.0) WRITE (6,54) NT,J
      GC TO 44
      IF (NPR.GT.0) WRITE (6,54) NT,J
      GC TO 44
      NT = NT+1
      AT = NT+1
      GC TO 22

      SUCCESS: WE HAVE A REPLACEMENT FOR VERTEX J.
25     FUN(J) = FUNTRY
      FUNCID = FUNTRY
      NPT = NPT+1

      EVERY 100 TH PERMISSIBLE TRIAL, RECCOMPUTE CENTROID SUMMATION TC
      AGAIN CREEPING ERROR.
      IF (MCD(NPT,100).NE.0) GO TO 39

      CC 38 I=I,NV
      SUM(I) = 0.

      CC 37 N=N,I,K
      SUM(I) = SUM(I)+V(I,N)

27     CONTINUE

      CEN(I) = SUM(I)/K
      CONTINUE

      LC = 0
      GC TO 41

28     CC 40 I=I,NV
      SUM(I) = SUM(I)+V(I,J)

      LC = J

      IF (NPR.LE.0) GO TO 42

```

EXPL5770	EXPL5780	EXPL5790	EXPL5800	EXPL5810	EXPL5820	EXPL5830	EXPL5840	EXPL5850	EXPL5860	EXPL5870	EXPL5880	EXPL5890	EXPL5900	EXPL5910	EXPL5920	EXPL5930	EXPL5940	EXPL5950	EXPL5960	EXPL5970	EXPL5980	EXPL5990	EXPL6000	EXPL6010	EXPL6020	EXPL6030	EXPL6040	EXPL6050	EXPL6060	EXPL6070	EXPL6080	EXPL6090	EXPL6100	EXPL6110	EXPL6120	EXPL6130	EXPL6140	EXPL6150	EXPL6160	EXPL6170	EXPL6180	EXPL6190	EXPL6200	EXPL6210	EXPL6220	EXPL6230	EXPL6240
----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------

```

C      IF (MOC(NFT,NPR).NE.0) GO TO 42
C      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,LC)
C      HAS THE MAX. NUMBER OF TRIALS BEEN REACHED WITHOUT CONVERGENCE?
C      IF (ACT, GC, TC NEW TRIAL.
C      42 IF (NT.GE.NTA) GO TO 43
C
C      NEXT-TC-WCRST VERTEX NOW BECOMES WORST.
C      J=JN
C      GC TO 15
C      43 IER = 3
C      IF (NPR.GT.0) WRITE (6,56)
C
C      COLLECTOR POINT FOR ALL ENDINGS.
C      1) CANNOT DEVELOP FEASIBLE VERTEX. IER = 1
C      2) CANNOT DEVELOP A NO. LARGER-WCRST VERTEX. IER = 2
C      3) FUNCTION VALUE UNCHANGED FOR K TRIALS. IER = 3
C      4) LIMIT ON TRIALS REACHED. IER = 4
C      5) CANNOT FIND FEASIBLE VERTEX AT START. IER = 5
C      44 CONTINUE
C
C      FIND BEST VERTEX.
C      CALL FBV (K,FUN,M)
C      IF (IER.GE.3) GO TO 46
C
C      RESTART IF THIS SOLUTION IS SIGNIFICANTLY BETTER THAN THE PREVIOUS,
C      CR IF THIS IS THE FIRST TRY.
C      IF (NPR.LE.0) GO TO 45
C      WRITE (6,57) (M,YMN,FUN(M))
C      45 IF (FUN(M).GE.YMN) GO TO 49
C      IF (ABS (FUN(M)-YMN).LE.AMAXI(EP,EP*YMN)) GC TO 45
C
C      GIVE IT ANOTHER TRY UNLESS LIMIT ON TRIALS REACHED.
C      46 YMN = FUN(M)
C      FUN(1) = FUN(M)
C
C      CC 47 I=1,NV
C      CEN(I) = V(I,M)
C      SUM(I) = V(I,M)
C      47 V(I,I) = V(I,M)
C
C      CC 48 I=1,NVT
C      XS(I) = V(I,M)
C
C      48 XS(I) = V(I,M)

```

```

BXPL6250
BXPL6260
BXPL6270
BXPL6280
BXPL6290
BXPL6300
BXPL6310
BXPL6320
BXPL6330
BXPL6340
BXPL6350
BXPL6360
BXPL6370
BXPL6380
BXPL6390
BXPL6400
BXPL6410
BXPL6420
BXPL6430
BXPL6440
BXPL6450
BXPL6460
BXPL6470
BXPL6480
BXPL6490
BXPL6500
BXPL6510
BXPL6520
BXPL6530
BXPL6540
BXPL6550
BXPL6560
BXPL6570
BXPL6580
BXPL6590
BXPL6600
BXPL6610
BXPL6620
BXPL6630
BXPL6640
BXPL6650
BXPL6660
BXPL6670
BXPL6680
BXPL6690
BXPL6700
BXPL6710
BXPL6720

```





```

C
C
SUBROUTINE FBV (K, FUN, M)
  DIMENSION FUN(50)
  M = 1
  DO 1 I=2, K
    IF (FUN(M).LE.FUN(I)) GO TO 1
    M = I
  1 CONTINUE
  RETURN
  END
C
C

```

```

FBV 10
FBV 20
FBV 30
FBV 40
FBV 50
FBV 60
FBV 70
FBV 80
FBV 90
FBV 100
FBV 110
FBV 120
FBV 130

```





```

C+++++ SUBROUTINE RCUTH
C+++++
C+++++ PURPOSE: TO DETERMINE IF A POLYNOMIAL IN S HAS RIGHT HALF
C+++++ PLANE ROOTS.
C+++++
C+++++ CALLING SEQUENCE:
C+++++ CALL RCUTH(Y,N,ISTABL)
C+++++
C+++++ DESCRIPTION OF PARAMETERS:
C+++++ Y = A ONE DIMENSIONAL ARRAY CONTAINING THE COEFFICIENTS
C+++++ CF THE POLYNOMIAL IN DESCENDING POWERS OF THE
C+++++ INDEPENDENT VARIABLE S.
C+++++ N = DIMENSION OF THE ARRAY Y.
C+++++ ISTABL = INTEGER OUTPUT; THE POLYNOMIAL HAS NO RIGHT HALF
C+++++ 0--- PLANE ROOTS.
C+++++ 1--- INDICATES THE POLYNOMIAL HAS RIGHT HALF
C+++++ PLANE ROOTS.
C+++++
C+++++ SLEROUTINES AND FUNCTION SUB-PROGRAMS REQUIRED:
C+++++ NCNE
C+++++
C+++++ METHOD: POLYNOMIAL IS CHECKED FOR RIGHT HALF PLANE ROOTS BY
C+++++ FORMING THE ROUTH ARRAY FROM THE POLYNOMIAL COEFFICIENTS
C+++++ INPUT TO THE SUBROUTINE.
C+++++
C+++++ SUBROUTINE RCUTH (Y,N,ISTABL)
C+++++ DIMENSION A(40,21), Y(1)
C C CC 2 I=1,40
C C CC 1 J=1,21
C C A(I,J)=0.0
C 1 CCONTINUE
C 2 CCONTINUE
C
C ISTABL = 0
C L=0
C KK = N/2-0

```

```

C      KK1 = KK+1
C      FF = KK1
C      K = KK*2.0
C      IF (K.EQ.0.N) GO TO 5
C      N2 = N-1
C      CC 3 J=1,N,2
C      L = L+1
C      A(L,L) = Y(J)
C      = CCNTINUE
C
C      CC 4 J1=2,N2,2
C      LL = LL+1
C      A(2,LL) = Y(J1)
C      IF (A(2,1).EQ.0.0) A(2,1)=0.0001
C      4 CCNTINUE
C
C      GC TO 8
C      5 KK1 = KK
C
C      CC 6 J=1,N,2
C      L = L+1
C      A(1,L) = Y(J)
C      6 CCNTINUE
C
C      CC 7 J1=2,N,2
C      LL = LL+1
C      A(2,LL) = Y(J1)
C      7 CCNTINUE
C
C      IF (A(2,1).EQ.0.0) A(2,1)=0.0001
C
C      CC 10 I=3,N
C
C      CC 5 N=1,KK
C      A(I,M) = (A(I-1,1)*A(I-2,M+1)-A(I-2,1)*A(I-1,1))/A(I-1,1)
C      5 CCNTINUE
C
C      IF (A(1,1).EQ.0.0) A(1,1)=0.0001
C      FF = FF-0.5
C      KK = FF
C      10 CCNTINUE
C
C      CC 11 I=1,N
C      IF (A(1,1).LT.0.0) ISTATB=1

```

ROUT 970  
ROUT 980  
ROUT 990  
ROUT 1000

11 CCNTINE  
RETURN  
ENC

C





```

      IF (I-IDIMX) 4,4,6
      IF (I-IDIMY) 5,5,7
      Z(I) = X(I) + Y(I)
      4  GO TO 8
      5  Z(I) = Y(I)
      6  GO TO 8
      7  Z(I) = X(I)
      8  CONTINUE
      9  ICIMZ = NDIM
      RETURN
      END

```

C

```

PADD 450
PADD 500
PADD 510
PADD 520
PADD 530
PADD 540
PADD 550
PADD 560
PADD 570
PADD 580
PADD 590
PADD 600

```

[illegible]

```

C AND FORM THE SUM OF THESE TO FIND THE TOTAL PHASE SHIFT INVOLVED.
C
CC 3 I=1,N-RI(I)
VI = CMEG-RI(I)
IF ((VR.EQ.0.0).AND.(VI.EQ.0.0)) GC TO 1
ANGR = ATAN2(VI,VR)
ANGC = ANGR*57.29578
GC TO 2
1 ANCD = 0.0
2 TANGD = TANGD+ANGC
3 CC CONTINUE
4 RETURN
C

```

```

CPHAS 490
PHAS 500
PHAS 510
PHAS 520
PHAS 530
PHAS 540
PHAS 550
PHAS 560
PHAS 570
PHAS 580
PHAS 590
PHAS 600
PHAS 610
PHAS 620
PHAS 630

```

```

C+++++SLBROUTINE SIMULA+++++
C+++++
C+++++PURPOSE: TO COMPUTE THE FREQUENCY RESPONSE OF THE OPEN LCAP
C+++++SYSTEM AFTER THE OPTIMIZED COMPENSATOR COEFFICIENTS
C+++++HAVE BEEN DETERMINED. THE FREQUENCY RESPONSE IS
C+++++COMPUTED OVER THE FREQUENCY RANGE SPECIFIED BY THE
C+++++USER, BUT AT A DENSER INTERVAL THAN IS NORMALLY USED.
C+++++IN THE OPTIMIZATION PART OF THE PROGRAM, NOT EE
C+++++TC SHOW ANY RESONANT PEAKS WHICH MIGHT FREQUENCIES
C+++++CBVIOUS IN THE GRAPHS IF THE DISCRETE
C+++++CHOSEN BY THE USER ARE WIDELY SPACED.
C+++++
C+++++CALLING SEQUENCE:
C+++++CALL SIMULA(MIN,WMAX)
C+++++
C+++++DESCRIPTION OF PARAMETERS:
C+++++WMIN = MINIMUM VALUE OF FREQUENCY RANGE BEING COMPLETED
C+++++WMAX = MAXIMUM VALUE OF FREQUENCY RANGE BEING COMPLETED
C+++++
C+++++SLBROUTINE AND FUNCTION SUBPROGRAMS REQUESTED
C+++++PVAL, CCNORM, PRQD, PRBM, PQPB, PHASE
C+++++
C+++++NOTE: THE MAGNITUDE AND PHASE VALUES COMPUTED ALONG WITH THE
C+++++FREQUENCIES AT WHICH THEY ARE COMPUTED ARE STORED IN THE
C+++++ARRAYS SAMAG, SAPHSC, AND WFREQS. THESE ARRAYS ARE
C+++++OUTPUT VIA THE CCNMCK STATEMENT.
C+++++
C+++++SLBROUTINE SIMULA (MIN,WMAX)
C+++++CCNORM=CCN,NTF,NIN,NO,NOEG,A(20),B(20),CMAG(500),DOPHSD(500),
C+++++1FREQ(500),NSN,NSD,COFF,NSN(40),COFFASD(40),NS,CMAG(500),DOPHSD(500),
C+++++2COFF(500),NSN,NSD(500),M1,N2,N3,COFF,PI(1),N5,1ZCCT,APAG(500),A(1),FILE(
C+++++3=20),FREQ(500),AMAG(500),NMINFS,COFF,WRK(40),1PLCT,WFREQS(500),SAPHSD
C+++++4500),SAMAG(500),DIFF1(500),DIFF2(500),ICLST,CFMRK(140)
C+++++CINFSICN RNR(40),RNI(40),RDR(40),RDI(40)
C+++++
C+++++C
C+++++C I=1,500
C+++++C FREQS(I)=0.0
C+++++C WFSAG(I)=0.0
C+++++C 1 CCNTINUE

```

```

C      N7 = NSN+1
C      N8 = NSD+1
C      XCMEG = 500.0
C      WFREQS(I) = WMIN
C      Y = (ALOG10(WMAX) - ALOG10(WMIN)) / (XCMEG - 1.0)
C      Z = 10.0**Y
C      CC 2 I=2,500
C      J = I-1
C      WFREQS(I) = WFREQS(J)*Z
C      CC 5 K=1,500
C      CALL PVAL (COFASN, NSN, 0.0, WFREQS(K), QNUMR, CNUMI)
C      CALL PVAL (COFASD, NSD, 0.0, WFREQS(K), QDENR, CDENI)
C      IF (SQRT(CDENI**2 + QDENR**2)) 4,3,4
C      SAMAG(K) = 1.E+30
C      CR(K) = 1.E+30
C      CI(K) = 1.E+30
C      CC 10 5
C      SAMAG(K) = SQRT(QNUMI**2 + QNUMR**2) / SQRT(CDENI**2 + CDENR**2)
C      IF (IZOF.EQ.1) SAMAG(K) = SAMAG(K) * ABS((SIN(PI*WFREQS(K)/WS)) / (PI*W
C      IFREQS(K)/WS))
C      CR(K) = (QNUMR*QDENR + QNUMI*QDENI) / (CDENR**2 + CDENI**2)
C      CI(K) = (QNUMI*QDENR - CNUMR*CDENI) / (CDENR**2 + CDENI**2)
C      CC CONTINUE
C      IF (N7.EQ.1) GO TO 7
C      CC 6 I=1,N7
C      CCFWK(I) = COFASN(I)
C      CC CONTINUE
C      CALL CONRM (COFWRK,N7,XNORM1)
C      CALL PRCD (COFWRK,N7,PAR,RNI,CFWK1,NRTSN1,IERN1)
C      IF (IERN1.EQ.0) GO TO 8
C      CALL PRFM (COFWRK,N7,PAR,RNI,CFWK1,NRTSN2,IERN2)
C      IF (IERN2.EQ.0) GO TO 8
C      WRITE (6,14)
C      WRITE (6,15)
C      WRITE (6,16)
C      WRITE (6,15)
C      WRITE (6,14)
C      CC 10 12
C      IF (N8.EQ.1) GO TO 10
C      CC 5 J=1,N8

```



## APPENDIX C

### DESCRIPTION OF FORTRAN VARIABLE NAMES

FORTRAN VARIABLE	UNIT	DESCRIPTION
A(I)	None	Array containing the coefficients of the plant transfer function numerator.
ADAMG(I)	db	Array containing the desired open loop magnitude response in db at the discrete frequencies specified by the designer.
AMAG(I)	db	Output array containing the magnitude of the open loop frequency response in db at the discrete frequencies specified.
ANGDEC	Degrees	Phase angle contribution of the open loop system transfer function denominator at a specific frequency.
ANGDEF	Radians	Phase angle contribution of the open loop system transfer function denominator at a specific frequency.
ANGNUL	Degrees	Phase angle contribution of the open loop system transfer function numerator at a specific frequency.
ANGNUR	Radians	Phase angle contribution of the open loop system transfer function numerator at a specific frequency.
E(I)	None	Array containing the coefficients of the plant transfer function denominator.
CO(I)	None	One dimensional working array used primarily in reading polynomial coefficients and inputting data to root finding subroutines.
COF(I)	None	One dimensional working array used primarily to store resulting coefficients used in checking accuracy of root finding subroutines.
COFASN(I)	None	Array containing the coefficients of the open loop transfer function numerator.



FORTRAN VARIABLE	UNIT	DESCRIPTION
COFASL (I)	None	Array containing the coefficients of the open loop system transfer function denominator.
COFWRK (I)	None	One dimensional working array used primarily to temporarily store polynomial coefficients in performing polynomial manipulations.
DELPHI	Degrees	Difference between two numerator phase angle contributions at successive discrete frequencies.
DELPSI	Degrees	Difference between two denominator phase angle contributions at successive discrete frequencies.
DIFF1 (I)	db	Difference between the desired and actual magnitude of the open loop frequency response at the discrete frequency values specified in the input.
DIFF2 (I)	Degrees	Difference between the desired and actual phase of the open loop frequency response at the discrete frequency values specified in the input.
DMAG (I)	None	Array containing the desired magnitude response at the discrete frequencies specified by the designer.
DQPHSL (I)	Degrees	Input array containing the desired phase profile of the open loop system at the discrete frequencies chosen.
DWFQ	Radians	Interval to be used if discrete frequency values are to be incremented linearly.
FREQ (I)	None	Logarithm of the discrete frequencies used in constructing abscissa for Bode plots.
GAIN	None	Gain of plant transfer function.
GAINCC	None	Gain of compensator transfer function.

FORTRAN VARIABLE	UNIT	DESCRIPTION
IDB	None	Input integer to be set equal to unity if the desired open loop magnitude response is input in db.
IERBXF	None	Output integer indicating the completion code of the minimization routine.
INFORM	None	Alphabetic input indicating if coefficients are to be read in factored or polynomial form.
IPLOT	None	Integer input, to be set to unity if CALCOMP plots of results are desired.
IZOH	None	Integer input set equal to unity if there is a zero order hold in the error signal path prior to the plant and compensator.
KNOW	None	Integer input indicating if discrete frequency values are to be read in or incremented linearly from the minimum frequency.
NEODE	None	Integer input to be set equal to unity if Bode plot is not desired.
NCD	None	Order of compensator transfer function denominator.
NCN	None	Order of compensator transfer function numerator.
NICHCL	None	Integer input to be set equal to unity if Nichol's plot is not desired.
NMINFS	None	Integer input to be set equal to unity if a nonminimum phase solution is to be allowed.
NOMEG	None	Integer specifying the total number of discrete frequency points being considered in the input frequency response profile. This number must be $\leq 500$ .
NSD	None	Order fo the system open lccp transfer function denominator.

FCRTRAN VARIABLE	UNIT	DESCRIPTION
NSN	None	Order of the system open loop transfer function numerator.
NTFD	None	Order of plant transfer function denominator.
NTFN	None	Order of plant transfer function numerator.
NYQST	None	Integer input to be set equal to unity if Nyquist plot is not desired.
N3	None	Dimension of array containing plant transfer function numerator coefficients.
N4	None	Dimension of array containing plant transfer function denominator coefficients.
QDENI	None	Value of the imaginary part of the system open loop transfer function denominator evaluated at a particular frequency.
QDENR	None	Value of the real part of the system open loop transfer function denominator evaluated at a particular frequency.
QMAG(I)	None	Output array containing the magnitude of the frequency response computed for the open loop system transfer function at the discrete frequencies specified.
QNUMI	None	Value of the imaginary part of the system open loop transfer function numerator evaluated at a particular frequency.
QNUMR	None	Value of the real part of the system open loop transfer function numerator evaluated at a particular frequency.
QPHSD(I)	Degrees	Output array containing the phase of the frequency response of the open loop system at the discrete frequencies specified.

FORTRAN VARIABLE	UNIT	DESCRIPTION
CPHSR (I)	Radians	Output array containing the phase of the frequency response of the open loop system transfer function at the discrete frequencies specified.
SAMAG (I)	db	Array containing the magnitude of the simulation of the resulting optimized system transfer function.
SAPHSL (I)	Degrees	Array containing the phase of the simulation of the resulting optimized system transfer function.
T	Secs	Sampling period if a zero order hold is present in the system.
WFREQ(I)	Radians	Array containing the discrete frequencies specified by the designer.
WMAX	Radians	Input specifying the maximum frequency value of the range of frequencies being considered.
WMIN	Radians	Input specifying the minimum frequency value of the range of frequencies being considered.
WS	Radians	Output giving the sampling frequency if a zero order hold is present in the system.
XS (I)	None	Array in which values to be varied by the minimization routine are stored. At the start this array contains the initial guess of the coefficient values and at completion the array contains the coefficient values that yield the minimum cost function.

# LIST OF REFERENCES

1. Massachusetts Institute of Technology, Electronic Systems Laboratory Report ESL-R-409, On the Design of a Digital Computer Program for the Design of Feedback Compensators in Transfer Function Form, by M. Athans, December 1969.
2. Beveridge, G. S. G. and Schechter, R. S., Optimization: Theory and Practice, p.453-456, McGraw-Hill, 1970.
3. Bode, H. W., Analysis and Feedback Amplifier Design, D. Van Nostrand Company, 1945.
4. Box, M. J., "A New Method of Constrained Optimization and a Comparison With Other Methods", Computer Journal, p.42-52, 8 April 1965.
5. Aerocspace Corporation Report TR-1001(2307)-20, The Application of Modern Computing Technology to Control System Analysis and Design Problems, by T. C. Coffey, June 1967.
6. Jacoby, S. L. S., Kowalik, J. S., and Fizzo, J. T., Iterative Methods for Nonlinear Optimization Problems, p.195-197, Prentice-Hall, 1972.
7. Kuo, P. F., and Kaiser, J. F., Editors, Systems Analysis by Digital Computer, Wiley, 1966.
8. Leondes, C. I., Editor, Control and Dynamic Systems Advances in Theory and Applications, v. 11, p.25-143, Academic Press, 1974.
9. Melsa, J. L. and Jones, S. K., Computer Programs for Computational Assistance in the Study of Linear Control Theory, 2d. ed., McGraw Hill, 1973.
10. Melsa, J. L. and Schultz, D. G., Linear Control Systems, p. 513-591, McGraw Hill, 1969.
11. NASA Contractor Report CR-2248, An Innovative Approach to Compensator Design, by Jerrell H. Mitchell and Willie I. McDaniel, Jr., May 1973.
12. Oldenburger, R., Editor, Frequency Response, The MacMillian Co., 1956.
13. Page, J. A., Automated Design of Feedback Control Systems, Ph.D. Thesis, University of California, Los Angeles, 1970.
14. Rosenbrock, H. H., Computer Aided Control System Design, Academic Press, 1974.
15. Sanathanan, C. K. and Tsukui, H., "Synthesis of Transfer Function from Frequency Response Data", International Journal of Systems Science, v. 5, no. 4, p.41-54, May 1974.

# LIST OF REFERENCES (cont.)

16. NASA Technical Memorandum X-1443, FORMAC Program to Assist in the Analysis of Linear Control Systems Using State Variable Feedback Design Technique, by C. R. Slivinsky, L. T. Delleher, and D. J. Arfasi, October 1967.
17. Thaler, G. J., Design of Feedback Systems, Dowden, Hutchinson, and Ross, 1973.
18. Thaler, G. J. and Brown, R. G., Analysis and Design of Feedback Control Systems, 2d. ed., McGraw Hill, 1960.
19. Truxal, J. G., Introductory System Engineering, McGraw Hill, 1972.
20. Office of Naval Research London Conference Report C-20-73, Conference on "Computer Aided Control System Design", by A. H. Waynick, 10 September 1973.
21. Computer Aided Control System Design Conference Publication Number 96, A Method for Linear Systems Order Reduction, by S. C. Chuang, p.213-218, 4 April 1973.

# INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2. Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3. Department Chairman, Code 62 Naval Postgraduate School Monterey, California 93940	2
4. Professor George J. Thaler, Code 62TR Naval Postgraduate School Monterey, California 93940	5
5. Lt. Anthony J. Mancini, USN 1514 Electric St. Scranton, Pa. 18509	3
6. Dr. Jerrel R. Mitchell Department of Electrical Engineering Mississippi State University Mississippi State, Mississippi 39762	1
7. Mr. Jay Cameron IBM Monterey and Cottle Roads San Jose, California 95114	1

8. Mr. Peter Gramata 1  
IBM  
Monterey and Cottle Roads  
San Jose, California 95114
9. Mr. Eckert McIntosh 1  
IBM  
Monterey and Cottle Roads  
San Jose, California 95114
10. Mr. Martin Dost 1  
IBM  
Monterey and Cottle Roads  
San Jose, California 95114
11. Mr. Ecn Palmer 1  
IBM  
Monterey and Cottle Roads  
San Jose, California 95114
12. Mr. W. E. Syn 1  
IBM  
Monterey and Cottle Roads  
San Jose, California 95114
13. Professor A. G. J. MacFarlane 1  
Dept. of Electrical Engineering  
University of Manchester  
Inst. of Science and Technology  
Manchester England 067609
14. Dr. Eui-Tien Rung 1  
Dept. of Applied Sciences  
Universite du Quebec a Chicoutimi  
930 est, rue Jacques-Cartier  
Chicoutimi, Quebec  
G7H 2E1



15. Mr. Dave Noble

1

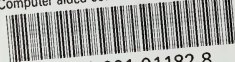
IBM

Monterey and Cottle Roads

San Jose, California 95114

thesM2786

Computer aided control system design usi



3 2768 001 01182 8

DUDLEY KNOX LIBRARY